On the evaluation and generalization of visual representations

Mert Bülent Sariyildiz

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : MSTII - Mathématiques, Sciences et technologies de l'information, Informatique
Spécialité : Informatique
Unité de recherche : Laboratoire Jean Kuntzmann

De l'évaluation et la généralisation des représentations visuelles

On the evaluation and generalization of visual representations

Présentée par :

Mert Bülent SARIYILDIZ

Direction de thèse :

Karteek ALAHARI Chargé de recherche HDR, INRIA Centre Grenoble-Rhône-Alpes	Directeur de thèse
Diane LARLUS Ingénieur Docteur, NAVER LABS Europe	Co-encadrante de thèse
Yannis KALANTIDIS Ingénieur Docteur, NAVER LABS Europe	Co-encadrant de thèse

Rapporteurs :

Matthieu CORD
Professeur des Universités, SORBONNE UNIVERSITE
Yannis AVRITHIS
Docteur en sciences HDR, IARAI

Thèse soutenue publiquement le **29 juin 2023**, devant le jury composé de :

Matthieu CORD Professeur des Universités, SORBONNE UNIVERSITE	Rapporteur
Yannis AVRITHIS Docteur en sciences HDR, IARAI	Rapporteur
Jocelyn CHANUSSOT Professeur des Universités, GRENOBLE INP	Président
Cordelia SCHMID Directeur de recherche, INRIA CENTRE DE PARIS	Examinatrice
Thomas MENSINK Scientist, Google	Examinateur

Invités :

Karteek ALAHARI
Chargé de recherche HDR, INRIA Centre Grenoble-Rhône-Alpes
Diane LARLUS
Ingénieur Docteur, NAVER LABS Europe
Yannis KALANTIDIS
Ingénieur Docteur, NAVER LABS Europe

Abstract

A primary goal of computer vision is to equip machines with the ability to extract information from visual data, such as images or videos, and thereby enable them to perform tasks defined on such data. While the specifics of the information to be extracted from data hinges on the task at hand, solving many complex tasks simultaneously necessitates a mechanism capable of extracting a comprehensive set of information from data. Therefore, substantial effort has been dedicated to the development of deep learning models capable of encoding such information into robust visual representations.

A prominent strategy in this context involves initially training a model on a large-scale dataset, such as ImageNet-1K, and subsequently employing this model for the task at hand (*e.g.*, image classification or object detection as downstream tasks on other datasets). In order to ensure that the model can successfully handle a variety of downstream tasks with minimal effort, the focus in this preliminary training phase is on learning image representations that can exhibit cross-task applicability, *i.e.*, transfer from the initial task to downstream tasks.

This thesis delves into learning transferable image representations by deep neural networks from three aspects. In the first part, we focus on evaluating the transferability of representations from the perspective of concept generalization, wherein the aim is to accurately recognize unseen concepts, *i.e.*, concepts not encountered during the model's training phase. We do this by proposing ImageNet-CoG, a benchmark including downstream tasks specifically designed for measuring concept generalization, and conducting an exhaustive evaluation of different representation learning methods on this benchmark. Our findings reveal that self-supervised methods are more resilient to concept generalization, *i.e.*, they learn more transferable representations for semantically less similar unseen concepts. Whereas, supervised methods tend to overfit more to the concepts seen during training, achieving better performance on seen concepts while learning less transferable representations for unseen ones.

Drawing from these observations, in the second part, we combine the strengths of supervised and self-supervised methods to maintain high performance on both seen concepts and downstream tasks. By adapting supervised methods with the techniques derived from the recent self-supervised methods, such as SimCLR and SwAV, we devise an improved training setup for supervised learning on ImageNet-1K. Models trained with our improved setup learn more transferable representations than the most recent self-supervised methods, when evaluated on a large collection of downstream image classification tasks. By further enhancing this setup with a prototype-based classification model, we achieve state-of-the-art

performance on ImageNet-1K (seen concepts) and the downstream tasks.

Lastly, in the third part, inspired by the recent surge in text-to-image generative models producing high-quality realistic images, we study whether such synthetic images allow training supervised models that can be utilized as effectively as models trained on real images. To investigate this, we generate synthetic clones of ImageNet-1K using Stable Diffusion and then train supervised models on these synthetic clones. Upon evaluating the resulting models on datasets comprising real images, we observe that training models on synthetic data leads to more transferable representations.

Keywords

Visual representation learning, computer vision, deep learning, artificial intelligence

Résumé

Un des objectifs principaux de la vision par ordinateur est de doter les machines de la capacité d'extraire des informations à partir de données visuelles, telles que les images ou les vidéos, leur permettant ainsi d'effectuer des tâches définies sur ces données. Bien que les informations à extraire de ces données dépendent fortement de la tâche à accomplir, la résolution simultanée de plusieurs tâches complexes nécessite un mécanisme capable d'extraire un ensemble complet d'informations à partir de ces données. Par conséquent, des efforts substantiels ont été consacrés au développement de modèles d'apprentissage profond capables d'encoder ces informations dans des représentations visuelles robustes.

Une stratégie de premier plan dans ce contexte consiste à entrainer un modèle initial sur un ensemble de données à grande échelle, tel que la base d'images ImageNet-1K, puis à utiliser ce modèle pour la tâche à accomplir (par exemple, la classification d'images ou la détection d'objets sur une autre base d'images). Afin de s'assurer de la capacité du modèle à gérer une variété de tâches cibles avec un minimum d'effort, l'accent est mis dans cette phase de pré-entrainement sur l'apprentissage de représentations d'images qui généralisent entre les tâches, c'est-à-dire qu'elles se transfèrent de la tâche initiale vers les tâches cibles.

Cette thèse se penche sur l'apprentissage de représentations d'images transférables par des réseaux de neurones profonds, et considère trois aspects. Dans une première partie, nous nous intéressons à l'évaluation de la transférabilité des représentations sous l'angle de la généralisation à de nouveaux concepts. L'objectif est de reconnaître des concepts non rencontrés lors de la phase d'apprentissage du modèle. Pour ce faire, nous proposons ImageNet-CoG, un 'benchmark' comprenant des tâches cibles spécifiquement conçues pour mesurer la généralisation d'un modèle à de nouveaux concepts. Nous procédons à une évaluation minutieuse de différentes méthodes d'apprentissage de représentations visuelle sur ce benchmark. Nos résultats révèlent que les méthodes auto-supervisées sont plus résiliantes à la généralisation à de nouveaux concepts, c'est-à-dire qu'elles apprennent des représentations plus transférables à des concepts non-observés au préalable et sémantiquement moins similaires. A l'inverse, les méthodes supervisées ont tendance à davantage sur-apprendre les concepts vus pendant l'entrainement, obtenant de meilleures résultats sur ceux-ci, mais apprenant des représentations moins transférables à de concepts nouveaux.

Partant de ce constat, dans une deuxième partie, nous combinons les atouts des apprentissages supervisé et auto-supervisé afin d'obtenir de bonnes performances à la fois sur les concepts de la tâche d'apprentissage mais aussi sur les tâches de transfert. En adaptant

les méthodes supervisées afin qu'elles utilisent des techniques empruntées aux méthodes auto-supervisées récentes, telles que SimCLR et SwAV, nous proposons une amélioration de l'apprentissage supervisé sur ImageNet-1K. Les modèles entrainés avec cette configuration améliorée apprennent des représentations plus transférables que les méthodes auto-supervisées les plus récentes, lorsqu'ils sont évalués sur une large collection de tâches cibles de classification d'images. En améliorant encore cette configuration avec un modèle de classification basé sur des prototypes, nous obtenons des performances état de l'art sur ImageNet-1K (concepts observés pendant l'apprentissage) ainsi que sur les tâches cibles.

Enfin, dans la troisième partie, inspirés par l'essor récent des modèles génératifs texte-image produisant des images réalistes de grande qualité, nous étudions si de telles images de synthèse permettent d'entraîner des modèles supervisés pouvant être utilisés à la place de modèles entraînés sur des images réelles. Pour étudier cela, nous générons des clones synthétiques d'ImageNet-1K à l'aide de l'outil Stable Diffusion, puis entrainons des modèles supervisés sur ces clones synthétiques. Lors de l'évaluation des modèles obtenus de cette façon sur des ensembles de données composés d'images réelles, nous observons que l'apprentissage de modèles à partir de données synthétiques produit des représentations plus transférables.

Mots clés

Apprentissage de représentations visuelles, vision par ordinateur, apprentissage profond, intelligence artificielle

Acknowledgments

This thesis would not have been possible without the support of many people. First and foremost, I would like to express my sincere gratitude to my supervisors, Karteek Alahari, Diane Larlus and Yannis Kalantidis, for being the greatest advisors. Their continuous guidance and support throughout my PhD have been instrumental not only in the contributions of this thesis but also in my growth as a researcher.

I would also like to thank my thesis committee members, Yannis Avrithis, Matthieu Cord, Jocelyn Chanussot, Thomas Mensink and Cordelia Schmid for their interest in my thesis, and kindly accepting to evaluate it.

I consider myself extremely lucky to have had the opportunity to be part of Inria Grenoble and Naver Labs Europe during my PhD. Both institutes have provided me with a super friendly, joyful and stimulating environment for research, among talented, ambitious and well-established researchers.

During my PhD journey, I have met many wonderful people who have been close friends, kind and helpful colleagues, invaluable sources of inspiration, or all three. I would like to extend my thanks to each one of them. Special mentions go to Noé Pion, Michel Aractingi, Assem Sadek, Juliette Bertrand, Juliette Marrie, Ginger Delmas, Pau de Jorge Aranda, Rafael Sampaio De Rezende, Riccardo Volpi, Tyler Hayes, Gustavo Rodrigues dos Reis, Romain Brégier, Thomas Lucas, Fabien Baradel, Philippe Weinzaepfel, Gregory Rogez, Boris Chidlovskii, Gabriela Csurka, Florent Perronnin, Julien Perez, Arnaud Sors, Christophe Legras, Irene Maxwell, Stefania Delassus, Cécile Paganelli, Yağmur Gizem Çınar, Emre Gülbahar, Nermin Samet, Gül Varol, Ahmet İşcen, Julien Mairal, Michael Arbel, Alexandre Zouaoui, Mathilde Caron, Valentin Gabeur, Romain Menegaux, Houssam Zenati, Minttu Alakuijala, Jules Bourcier, Nathalie Gillot and many others.

Finally, I am deeply grateful to my family, especially, my dear mom Dilek Atalay and wife Merve Sariyildiz, for their unconditional love and support, and all the sacrifices they have made on behalf of my thesis. I dedicate this thesis to my wife, our little flat in Grenoble and our Teddy.

Contents

Abstract i

Résumé iii

Acknowledgments v

1 Introduction 1
- 1.1 Challenges . . . 3
- 1.2 Contributions . . . 7
- 1.3 Publications . . . 10

2 Related work 13
- 2.1 Image representations . . . 13
  - 2.1.1 Hand-crafted representations . . . 14
  - 2.1.2 Representations learned with neural networks . . . 16
- 2.2 Forms of supervision to learn representations . . . 19
  - 2.2.1 Supervised learning . . . 20
  - 2.2.2 Self-supervised learning . . . 23
- 2.3 Learning representations from synthetic data . . . 24
- 2.4 Evaluating the generalization of representations . . . 25
- 2.5 Positioning the contributions . . . 28

3 Measuring concept generalization in visual representation learning 31
- 3.1 Introduction . . . 32
- 3.2 Related work . . . 35
- 3.3 The ImageNet-CoG benchmark . . . 36
  - 3.3.1 Seen concepts . . . 37
  - 3.3.2 Selecting eligible unseen concepts . . . 37
  - 3.3.3 Concept generalization levels . . . 38
  - 3.3.4 Evaluation protocol . . . 40
    - 3.3.4.1 Phase 1: Feature extraction . . . 41
    - 3.3.4.2 Phase 2: Evaluation . . . 41
    - 3.3.4.3 Metrics and implementation details . . . 42
- 3.4 Evaluating models on ImageNet-CoG . . . 42

3.4.1 Models . . . 42
3.4.2 Results . . . 45
3.4.2.1 Generalization to unseen concepts . . . 45
3.4.2.2 How fast can models adapt to unseen concepts? . . . 49
3.5 Further analysis on ImageNet-CoG . . . 50
3.5.1 What if we fine-tuned the backbones? . . . 50
3.5.2 word2vec as an alternative semantic similarity . . . 51
3.5.3 Potential label noise in ImageNet-CoG . . . 54
3.6 Conclusion . . . 54

4 Improving the generalization of supervised learning models 57
4.1 Introduction . . . 58
4.2 Related work . . . 60
4.3 An improved training setup for supervised learning . . . 62
4.4 Experiments . . . 64
4.4.1 Analysis of component design and hyper-parameters . . . 67
4.4.2 Analysis of learned features, class weights and prototypes . . . 70
4.4.3 Pushing the envelope of training-versus-transfer performance . . . 72
4.4.4 Evaluations on the additional transfer datasets . . . 74
4.5 Conclusion . . . 75

5 Learning transferable representations from synthetic ImageNet clones 77
5.1 Introduction . . . 79
5.2 Related work . . . 81
5.2.1 Learning with synthetic data . . . 81
5.2.2 Distillation of datasets and models . . . 82
5.3 Preliminaries . . . 83
5.4 Generating synthetic ImageNet clones . . . 84
5.4.1 Generating datasets using class names . . . 84
5.4.2 Addressing issues with semantics and domain . . . 86
5.4.3 Increasing the diversity of generated images . . . 86
5.5 Experiments . . . 88
5.5.1 Results on ImageNet datasets . . . 89
5.5.2 Resilience to domain shifts . . . 91
5.5.3 Transfer learning . . . 92
5.5.4 Impact of guidance scale and diffusion steps . . . 94
5.5.5 Analysis of the learned features . . . 94
5.6 Discussion . . . 97
5.7 Conclusions . . . 98

6 Conclusion **99**
6.1 Summary of contributions 99
6.2 Perspectives for future work 101

A ImageNet-CoG with the ImageNet 2021 release **105**

B ImageNet-CoG extended results **107**

C Hyper-parameters for training t-ReX models on IN-1K **117**

D Results per dataset for t-ReX models **119**

E Extended qualitative results for synthetic ImageNet clones **121**
E.1 Semantic errors 121
E.2 NSFW content 122
E.3 Misrepresentation of biodiversity 122
E.4 Semantic issues arising with backgrounds 123
E.5 Issues with diversity 124
E.6 Non-natural images 124
E.7 Varying the stable diffusion parameters 124

List of Figures

1.1 **Illustration on the ability of deep neural networks to learn hierarchical representations of images.** Given input images composed of pixels, low-level representations (*e.g.*, edges) are encoded first, which are combined to form higher-level representations (*e.g.*, object parts) that can be used to solve the task at hand, *e.g.*, image classification. Figure adapted from Goodfellow et al. [2016]. The image on the left courtesy of Merve Sariyildiz. 2

1.2 **Five hundred images randomly sampled from ImageNet-1K** [Deng et al. 2009, Russakovsky et al. 2015]. In this thesis, we mainly consider models trained on this dataset. Then, we evaluate the quality of their representations by transfering them to a variety of other datasets, including the ImageNet-CoG dataset that we propose in Chapter 3. 3

1.3 **Schematic illustration of the three contributions presented in this thesis**. (**a**) *Evaluation aspect:* In Chapter 3, we propose a new benchmark (ImageNet-CoG) tailored for evaluating the concept generalization performance of the models trained on ImageNet-1K. (**b**) *Modeling aspect:* In Chapter 4, we propose an improved training setup for training supervised models on the real images of ImageNet-1K. (**c**) *Data aspect:* In Chapter 5, we generate synthetic clones of ImageNet-1K to train supervised models. Note that the illustration is only for a high-level overview, and it omits many technical details. 8

2.1 **Illustration of image features** extracted by (**a**) handcrafted techniques and (**b**) deep neural networks. 14

2.2 **Illustrations of the various data augmentation operations.** Given an example image in (**a**), the label-preserving augmentations from (**b**) to (**j**) are: (**b**) Crop + resize, (**c**) Crop + resize + flip, (**d**) Color distortion (channel drop), (**e**) Color distortion (jitter), (**f**) Rotation, (**g**) Cutout, (**h**) Gaussian noise, (**i**) Gaussian blur and (**j**) Sobel filtering. Provided another image in (**k**), semantic transformations of (**a**) and (**k**) are (**l**) MixUp and (**m**) CutMix. Each augmentation operation has one or more parameters determining the output image. The images in (**a**) to (**j**) are taken from [Chen et al. 2020a], and the image in (**k**) is from ImageNet [Russakovsky et al. 2015]. 18

2.3 **Synthetic images generated by Imagen** for the given textual prompts. Figure taken from Saharia et al. [2022]. 25
2.4 **Illustration on the several aspects of generalization in computer vision**. All panda images are from Russakovsky et al. [2015], except the sketch image of a panda in (**a**), which is from Wang et al. [2019]. 26

3.1 **An overview of our Concept Generalization (CoG) benchmark.** (a) An example of five concepts from the ImageNet-21K dataset [Deng et al. 2009] (IN-21K), ranked by increasing *semantic* distance (decreasing Lin similarity [Lin 1998]) to the ImageNet-1K (IN-1K) dataset [Russakovsky et al. 2015] concept "Tiger cat". (b) We rank the 21K concepts of IN-21K according to their semantic distance to the 1000 concepts of IN-1K and split the ranked list to extract 5 groups of 1000 concepts. We refer to the five IN-1K-sized datasets of increasing semantic distance from IN-1K as *concept generalization levels*, denoted as $L_{1/2/3/4/5}$. (c) The proposed ImageNet-CoG benchmark uses a model trained on IN-1K as a feature extractor and evaluates its concept generalization capabilities by learning linear classifiers for each level of more and more challenging unseen concepts. 34
3.2 **Concept generalization levels.** We rank all the 5146 eligible IN-21K unseen concepts with respect to their similarity to IN-1K using Equation (3.2) and split the ranked list into 5 groups of 1000 concepts each. Each group defines a concept generalization level, each denoted by $L_{1/2/3/4/5}$. Gray-shaded areas correspond to concepts that are ignored. 39
3.3 The number of images per concept for IN-1K [Russakovsky et al. 2015] and each of the concept generalization levels obtained by Lin similarity. We end up with 1.17M, 1.17M, 1.15M, 1.16M, 1.14M images in total for levels $L_{1/2/3/4/5}$ respectively. Note that IN-1K has 1.33M images in total. 46
3.4 **Linear classification on ImageNet-CoG.** Top-1 accuracies for all the 31 models listed in Table 3.2 after training logistic regression classifiers on IN-1K and each level $L_{1/2/3/4/5}$. (a) Absolute Top-1 accuracy on all levels. (b)-(e) accuracy relative to the baseline ResNet50 for all the models, split across the four model categories presented in Section 3.4.1. 47
3.5 **Few-shot linear classification on ImageNet-CoG.** Top-1 accuracies for a subset of the models listed in Table 3.2 after training logistic regression classifiers on L_1, L_3, L_5 using $N = \{2, 4, 8, 16, 32, 64, 128\}$ training samples per concept. Performance when using all the samples is also shown for reference. (a)-(c): Absolute Top-1 accuracy. (d)-(f) accuracy relative to the baseline ResNet50. *The complete set of results for all the 31 models and levels is in the supplementary material.* 49

3.6 Comparison of training linear classifiers on **pre-extracted features** *vs.* **fine-tuning** backbones on each level. Y-axis shows the Top-1 accuracies obtained **relative** to the accuracy of the fine-tuned models. 51
3.7 **Semantic similarities** of the concepts captured by (i) Lin similarity [Lin 1998] on WordNet graph [Miller 1995] and (ii) cosine similarity of word2vec embeddings [Yamada et al. 2020] extracted from textual descriptions of concepts, *vs.* **visual similarities** encoded by ResNet50, on IN-1K and generalization levels $L_{1/2/3/4/5}$ of ImageNet-CoG. We report the performance of linear logistic regression classifiers trained on features extracted from the global average pooling layer of ResNet50. The orange line shows results obtained on 1000 *random* unseen concepts (line represents the mean accuracy obtained over 15 random splits). 53
3.8 **Illustration of the label noise in ImageNet-CoG.** 54

4.1 **Our proposed supervised learning setup** borrows multi-crop [Caron et al. 2020] and projectors [Chen et al. 2020a] from SSL to train on IN-1K (*top*). The projector g is discarded after training, and the ResNet backbone f is used as a feature extractor in combination with a linear classifier trained for each task, *e.g.* for texture classification on DTD [Cimpoi et al. 2014] (*bottom*). . 61
4.2 Architecture of the projector g_ϕ. 63
4.3 **The supervised models** we train using our proposed setup. I_g and $I_{g,l}$ represent only global crops or both global and local crops. 65
4.4 **Impact of the number of local crops** (M_l) on the performance on IN-1K (*left*) and transfer datasets (*right*) when varying the number of hidden layers (L) in the projector. The number of global crops (M_g) is 1 in all cases. . . . 67
4.5 Average intra-class ℓ_2-distance between samples from the same class (*top*) and sparsity as the percentage of feature dimensions close to zero (*bottom*), on IN-1K and averaged over transfer datasets. Gray and Orange arrows denote changes due to adding multi-crop and projectors, respectively. Best viewed in color. 68
4.6 (a) Average coding length per sample [Yu et al. 2020] over all *transfer* datasets. (b) Singular values across dimensions, averaged over the transfer datasets. We show the first 1000 dimensions (of 2048) for clarity. (c) Average similarity between class weight gradients $\nabla_{w_c}\mathcal{L}_{\text{CE}}$ during training. (d) Change in class weights W and prototypes U at every iteration across all classes (see text for details) for models trained using Equation (4.1) and Equation (4.2), respectively. Best viewed in color. 69

4.7 **Comparison on the training task vs transfer task performance for ResNet50.** We report IN-1K (Top-1 accuracy) and transfer performance (log odds) averaged over 13 datasets (5 ImageNet-CoG levels, Aircraft, Cars196, DTD, EuroSAT, Flowers, Pets, Food101 and SUN397) for a large number of our models trained with the supervised training setup presented in Section 4.3. Models on the convex hull are denoted by stars. We compare to the following state-of-the-art (SotA) models: Supervised: RSB-A1 [Wightman et al. 2021], SupCon [Khosla et al. 2020], SL-MLP [Wang et al. 2022a] and LOOK [Feng et al. 2022] with multi-crop; self-supervised: DINO [Caron et al. 2021]; semi-supervised: PAWS [Assran et al. 2021]. 71

5.1 **Overview of our experimental protocol.** During training, the model has access to synthetic images generated by the Stable Diffusion model, provided with a set of prompts per class. During evaluation, real images are classified by the frozen model. 78

5.2 **ImageNet-1K vs ImageNet-1K-SD.** The blue polygon shows the performance of a model trained on ImageNet-1K. The red polygon depicts the performance of one trained on ImageNet-1K-SD, *i.e.*, only on synthetic data generated with Stable Diffusion [Rombach et al. 2022] using the class names of ImageNet-1K. We report Top-5 accuracy for ImageNet test sets, and average Top-1 for transfer tasks. 80

5.3 **Qualitative results.** (A) Real ImageNet images. (B)-(G) Synthetic ImageNet-SD images generated with different prompts. Despite high photo-realistic quality, some issues are noticeable for (B) such as semantic errors *e.g.*, for the class “papillon”, lack of diversity, and distribution shifts *e.g.*, towards cartoons for the “pirate" class. Such issues are addressed with more expressive prompts in (C)-(D). 85

5.4 **Scaling the number of training images.** Average Top-1 accuracy on 10 transfer datasets (from Table 5.3) when training on ImageNet-100 using $(1/10)$-th to $50\times$ images (relative to the real dataset size). 93

5.5 **Impact of the guidance scale parameter and number of diffusion steps.** Top-1 accuracy on ImageNet-100 and averaged over 10 transfer datasets (from Table 5.3) for $p_c =$ “c, d_c”. In the left plot, steps are set to 50, in the right plot guidance scale is 7.5. 94

5.6 **Feature analyses** for models. We perform these analyses on top of features extracted from pretrained encoders f trained on either real or synthetic data for ImageNet-100 (training data is specified in the legends of the subfigures). For the purpose of this study, we use synthetic data generated with guidance scale equal to 7.5. *Sparsity* is measured by the percentage of dimensions close to zero [Kornblith et al. 2021]. *Intra-class ℓ_2-distance* is the average pairwise ℓ_2-distance between samples from the same class. These two metrics are computed on ℓ_2-normalized features. *Feature redundancy* [Wang et al. 2022b] is obtained by $\mathcal{R} = \frac{1}{d^2}\sum_i \sum_j |\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})|$, where $\mathbf{X} \in \mathbb{R}^{N \times d}$ is a feature matrix containing N samples, each encoded into a d-dimensional representation (2048 in our case) and $\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})$ is the Pearson correlation between a pair of feature dimensions i and j. *Coding length* [Yu et al. 2020] is measured by $R(\mathbf{X}, \varepsilon) = \frac{1}{2}\log\det(\mathbf{I}_d + \frac{d}{N\varepsilon^2}\mathbf{X}^\top\mathbf{X})$, where $\mathbf{I}_d$ is a d-by-d identity matrix, ε^2 is the precision parameter set to 0.5. 95

A.1 **Linear classification on ImageNet-CoG using blurred images for IN-1K.** Top-1 accuracies for all the 31 models listed in Table 3.2, after training logistic regression classifiers on the **blurred version** of IN-1K (IN-1K-b in the plots) and each level $L_{1/2/3/4/5}$. (a) Absolute Top-1 accuracy on all levels. (b)-(e) accuracy relative to the baseline ResNet50 for all the models, split across the four model categories presented in Section 3.4.1. 106

B.1 **Few-shot linear classification on ImageNet-CoG.** Top-1 accuracy for each method using logistic regression classifiers. We train them on pre-extracted features for the concepts in IN-1K and our generalization levels ($L_{1/2/3/4/5}$), with a few training samples per concept, *i.e.*, $N = \{1,2,4,8,16,32,64,128\}$. "All", the performance when all the samples are used, is also shown for reference. 108

B.2 **Relative few-shot linear classification on ImageNet-CoG.** The scores shown in Figure B.1 from a different perspective: all scores are **relative to ResNet50**. 110

E.1 **Qualitative results as we change the guidance scale parameter and the number of diffusion steps during Stable Diffusion generation.** The seed is fixed to `1947262` and the prompt is "robin, American robin, Turdus migratorius". Unless otherwise stated the scale (resp. steps) parameters are set to 7.5 (resp. 50). 126

E.2 **Qualitative results for class "Shih-Tzu"** to illustrate domain and diversity issues. Guidance scale is equal to 7.5. 127

E.3 **(cont.) Qualitative results for class "Shih-Tzu" to illustrate domain and diversity issues.** 128
E.4 **Qualitative results for classes "Rock crab" (left) and "Fiddler crab" (right)**, to illustrate issues around fine-grained and domain specific semantics. Guidance scale is equal to 7.5. 129
E.5 **Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". 130
E.6 **(cont.) Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". 131
E.7 **(cont.) Visualization of the images for the 100 ImageNet-100 classes** in the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". . . . 132
E.8 **(cont.) Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". 133
E.9 **(cont.) Visualization of the 100 ImageNet-100 classes** for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". 134

List of Tables

3.1 WordNet IDs of the 70 concepts considered problematic, therefore removed from the eligible list of unseen concepts. 38
3.2 **List of models evaluated on ImageNet-CoG.** 43
3.3 Unique architectures used by the models we evaluate on ImageNet-CoG, and the dimensionality of the feature vectors we extract from these architectures. 45

4.1 **Impact of the projector size** on performance, via the number of hidden layers L (*left*) and hidden units d_h (*right*). The default configuration: L=1, d_h=2048, d_b=256 and with ℓ_2-normalization of the input (highlighted rows). We use $M_g = 1$ and $M_l = 8$ ("Base+Mc"). 67
4.2 **Results on IN-1K concepts.** For each model, we report results on the IN-1K "Val" set (the x-axis of Figure 4.7), as well as on the test sets of IN-1K-v2 [Recht et al. 2019], IN-1K-sketch [Wang et al. 2019], IN-1K-R [Hendrycks et al. 2021a] and IN-1K-A [Hendrycks et al. 2021b], using in all cases the encoder and the linear classifier trained on the original IN-1K training set. IN-1K-v2 numbers are averaged over the three test sets (matched-frequency, threshold-0.7 and top-images). 74
4.3 **Transfer results on long-tail classification.** For each model, we train linear classifiers on the iNaturalist 2018 and iNaturalist 2019 datasets [Van Horn et al. 2018] with class-imbalanced data, following the LogReg protocol from ImageNet-CoG. 75

5.1 **Impact of data-augmentation** for models trained on real and synthetic datasets. Performance is measured on the validation set of ImageNet-100, *i.e.*, on real images. 89

5.2 **Results on ImageNet datasets.** Top-1 and Top-5 accuracy on several ImageNet datasets, namely IN-Val (the ILSVRC-2012 validation set [Russakovsky et al. 2015]), IN-v2 [Recht et al. 2019], IN-Sketch [Wang et al. 2019], IN-R [Hendrycks et al. 2021a] and IN-A [Hendrycks et al. 2021b]. In all cases, testing is done on real images. For the prompts, h_c (d_c) refers to the hypernym (definition) of class c provided by WordNet [Miller 1995], while b to scene classes from Places 365 [Zhou et al. 2017]. *IN-R and IN-A only cover a subset of the ImageNet-100 classes and we compute the reported metrics only on the common classes. Brick-colored scores denote performance higher than the models trained on real images. *Italics* denote results from models trained using real images. 90

5.3 **Top-1 accuracy on ten transfer learning datasets** for encoders trained on real and synthetic images. We treat encoders as feature extractors and train linear classifiers on top for each dataset. Brick-colored scores denote performance higher than the models trained on real images. We make the remarkable observation that representations from models trained on synthetic data can match the generalization performance of representations from models trained on millions of real images. *Italics* denote results from models trained using real images. 91

5.4 **Top-1 accuracy on the ImageNet-CoG benchmark [Sariyildiz et al. 2021].** We report performance for the best ImageNet-1K-SD model from Table 5.3 (with guidance scale equal to 2), and compare it to the state-of-the-art supervised and self-supervised models trained on the real images of ImageNet-1K, RSB-A1 [Wightman et al. 2021] and DINO [Caron et al. 2021], respectively. 93

A.1 **Comparison of the number of images in IN-1K and IN-1K-blurred.** . . 105

B.1 **Top-1 accuracies obtained by linear classifiers on IN-1K.** Table view corresponding to the 1st row in Figure B.1. 109

B.2 **Top-1 accuracies obtained by linear classifiers on L_1.** Table view corresponding to the 2nd row in Figure B.1. 111

B.3 **Top-1 accuracies obtained by linear classifiers on L_2.** Table view corresponding to the 3rd row in Figure B.1. 112

B.4 **Top-1 accuracies obtained by linear classifiers on L_3.** Table view corresponding to the 4th row in Figure B.1. 113

B.5 **Top-1 accuracies obtained by linear classifiers on L_4.** Table view corresponding to the 5th row in Figure B.1. 114

B.6 **Top-1 accuracies obtained by linear classifiers on L_5.** Table view corresponding to the last row in Figure B.1. 115

C.1 **Hyper-parameters for training** our models with ResNet50 architecture on IN-1K. Hyper-parameters shared by all models are given on the top part while the ones specific to **t-ReX** and **t-ReX*** are shown on the bottom part. 118

D.1 **Top-1 linear logistic regression accuracy per dataset.** Mean LO is average log odds computed over all transfer datasets (*i.e.* all datasets except IN-1K). In Section 4.4, we only plot IN-1K and Mean LO scores for each model. We repeat each evaluation 5 times with different seeds; variance is generally negligible. 119

Chapter 1

Introduction

Contents

1.1 Challenges 3
1.2 Contributions 7
1.3 Publications 10

In computer vision, many tasks require a semantic understanding of an image, for instance, classification, retrieval, detection, segmentation, captioning, or visual question answering. Models that tackle any of these tasks often consist of two main components: a primary mechanism to extract information from images and a secondary mechanism to perform the task based on the provided information. This notion of information extracted from an image (or any type of data, in general) is called the *feature* or *representation* of the image. Finding "good" image representations is an important concern in computer vision which ultimately impacts the performance of models [Goodfellow et al. 2016].

Before deep learning research became mainstream, researchers have extensively studied this problem of finding a good image representation via *handcrafted* approaches [Csurka et al. 2004, Lowe 2004, Perronnin and Dance 2007], where representations are designed by humans based on cues such as edges or corners in an image that are obtained by low-level image statistics like pixel gradients. With the advances in deep learning, we have seen a paradigm shift, from manually designing representations to designing neural networks to automatically *learning* them [Bengio 2012]. Deep neural networks can build a hierarchy of representations for their input, ranging from low-level features such as edges and corners to high-level concepts such as object parts, as illustrated in Fig. 1.1. Thanks to this ability of deep networks, for a given task to solve, it is possible to learn representations specific for this task from data. Moreover, such representations can be used across different tasks, which allows transfer of knowledge acquired by solving one task to other related tasks [Razavian et al. 2014]. This is a lucrative property, as it allows for solving a wide range of tasks with a single representation [Bommasani et al. 2021].

In this thesis, we study image representations learned by neural networks from three aspects. First, the *modeling* aspect: we aim to develop models that learn image representations suitable for solving not only the task which produces representations (the training task), but also

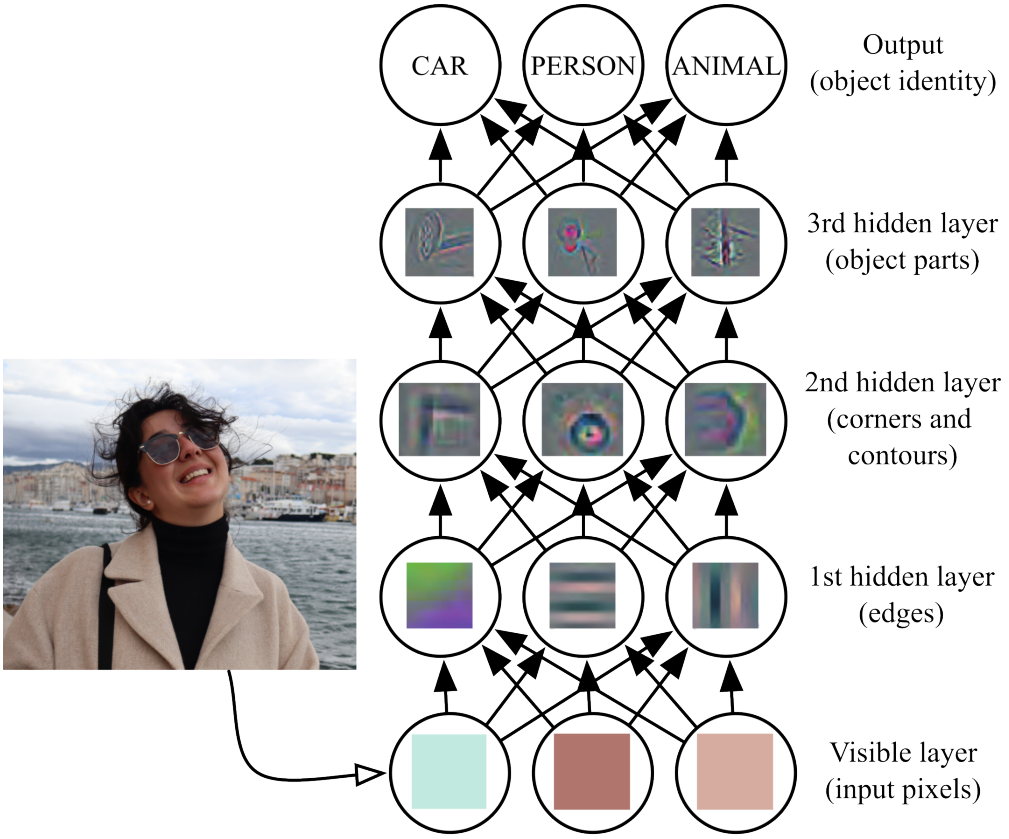


Figure 1.1: Illustration on the ability of deep neural networks to learn hierarchical representations of images. Given input images composed of pixels, low-level representations (*e.g.*, edges) are encoded first, which are combined to form higher-level representations (*e.g.*, object parts) that can be used to solve the task at hand, *e.g.*, image classification. Figure adapted from Goodfellow et al. [2016]. The image on the left courtesy of Merve Sariyildiz.

other tasks we might be interested in solving with the learned representations afterwards (often called downstream or transfer tasks). Second, the *data* aspect: we investigate the role of training data in learning such all-purpose representations. Third, the *evaluation* aspect: we focus on selecting the right transfer tasks to use, *i.e.*, to test the learned representations on, for reliably evaluating the quality of learned representations. In the remainder of this chapter, we discuss some of the challenges in these three axes (Sec. 1.1), then we pose a research question and present our contribution for each of them (Sec. 1.2). The list of papers published as part of this thesis is given at the end of this chapter (Sec. 1.3).

Let us first set the context for the specific problems we tackle in this thesis.

1.1 Challenges

Image representation learning by neural networks is a longstanding problem [Goodfellow et al. 2016]. Yet, research in this field has flourished especially after the introduction of AlexNet [Krizhevsky et al. 2012], a deep neural network that won the ImageNet Large-Scale Visual Recognition Challenge (a task of classifying images belonging to one thousand fine-grained object categories) in 2012. The dataset used in this challenge, also called ImageNet-1K [Russakovsky et al. 2015], has become one of the most established datasets in computer vision, and is used broadly by the community for large-scale evaluations in many fields including, but not limited to, learning image representations [Gidaris et al. 2018, Huh et al. 2016], developing generative models [Brock et al. 2019], designing network architectures [Elsken et al. 2019] or their training strategies [Cubuk et al. 2019]. A few randomly sampled images from ImageNet-1K are shown in Fig. 1.2.

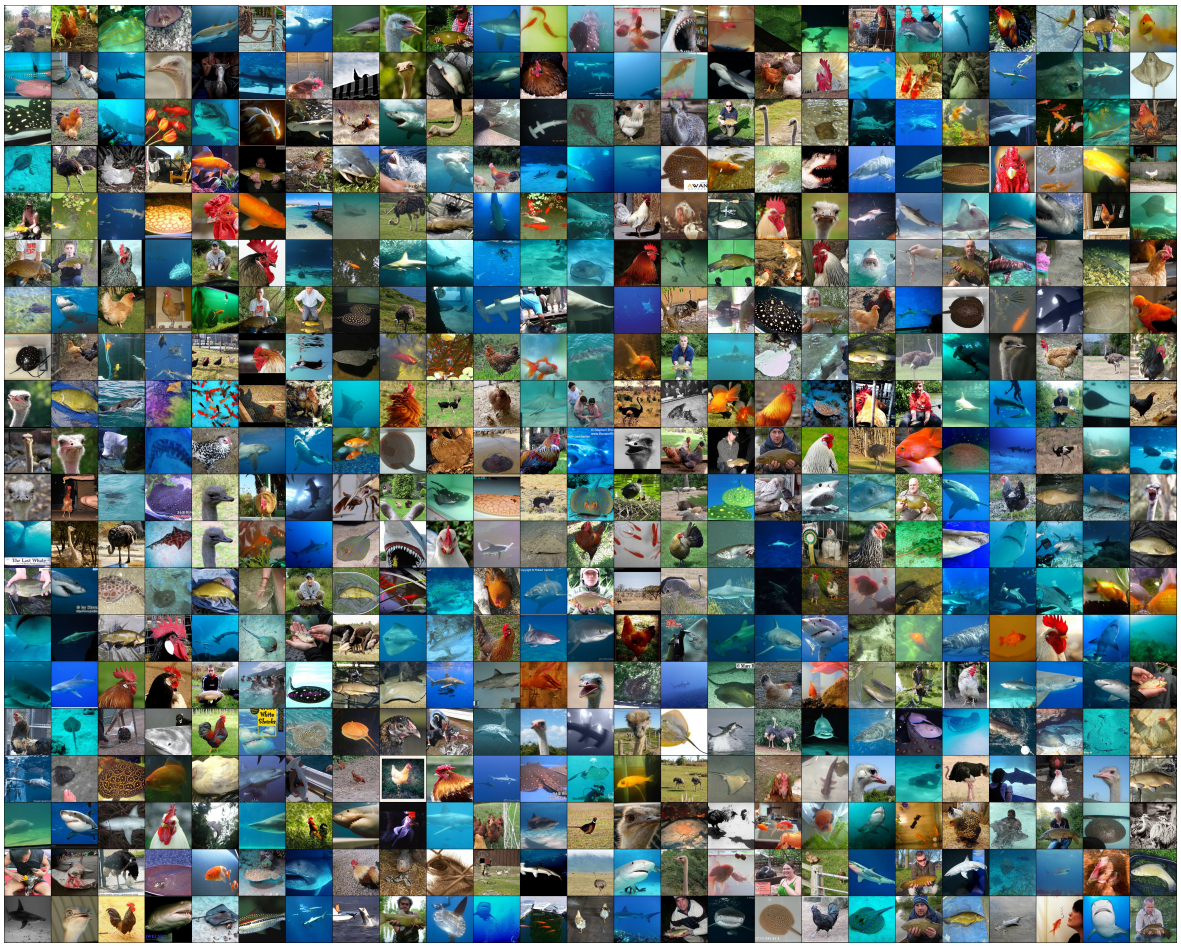


Figure 1.2: Five hundred images randomly sampled from ImageNet-1K [Deng et al. 2009, Russakovsky et al. 2015]. In this thesis, we mainly consider models trained on this dataset. Then, we evaluate the quality of their representations by transfering them to a variety of other datasets, including the ImageNet-CoG dataset that we propose in Chapter 3.

Soon after AlexNet's success, researchers have started to investigate the properties of the representations learned by deep neural networks on ImageNet-1K, and it turned out that these representations are not only useful for solving the original classification task, but also for other related tasks [Girshick et al. 2014, Razavian et al. 2014]. This ability of transferring representations across tasks (*i.e.*, transfer learning) opens the door to a much more ambitious goal of learning "all-purpose" [Oquab et al. 2023, Radford et al. 2021a, Yuan et al. 2021b] representations, *i.e.*, representations which are generic enough to be used for many (if not all) computer vision tasks. Although arguable, it is a reasonable goal from the cognitive science perspective. For instance, we, as humans, can handle a wide range of visual tasks. Moreover, for the tasks we are not familiar with, we can learn to perform them rather quickly by relying on our past experience. Similarly, achieving this goal in the context of neural networks would have positive implications, such as learning new tasks more efficiently (in terms of data or compute costs) and more accurately (with potentially improved performance on the whole repertoire of tasks).

Both ImageNet-1K and transfer learning are central components of this thesis. We specifically target the following three challenges in evaluating the transferability of representations learned on ImageNet-1K (the first) and learning better representations on ImageNet-1K that transfer to other tasks (the second and the third):

1. Measuring concept generalization in visual representation learning,
2. Improving the generalization of supervised learning models and
3. Learning transferable representations from synthetic ImageNet clones

Measuring concept generalization in visual representation learning

Given a model trained on ImageNet-1K, how can we assess whether its learned representations are suitable for solving other tasks? In other words, how can we evaluate the learned representations in terms of their generalization (transfer) capability to other tasks? The answer depends on which particular aspects of generalization we are interested in measuring, such as generalization to different input domains [Csurka et al. 2004] (*e.g.*, from natural images to sketches), different tasks [Zamir et al. 2018] (*e.g.*, from image classification to image segmentation), and different concepts [Lampert et al. 2009] (*e.g.*, from classifying cats *vs.* cars to classifying dogs *vs.* trucks). Measuring each of them has its own challenges.

In Chapter 3, we focus on generalization across concepts, where the goal is to transfer knowledge acquired on a set of seen concepts to newly encountered unseen concepts as effectively as possible. More precisely, we are interested in understanding what the challenges are and what kind of models or model architectures are better at this particular generalization aspect. Conventional wisdom attributes the main difficulty of concept generalization to the fact that

semantic relationships between seen and unseen concepts impact the affordance of knowledge transfer between them, when the other aspects of generalization are fixed, *e.g.*, using images from the same input domain (*e.g.*, natural images) and performing the same task (*e.g.*, image classification). This is usually based on a chain of thought following two observations. First, visual similarity of concepts is correlated with their semantic similarity [Deselaers and Ferrari 2011], that is, if two concepts are semantically similar, they are visually similar as well (*e.g.*, cats and dogs). Second, it is easier to transfer knowledge across visually more similar concepts [Huh et al. 2016]. For instance, it is reasonable to expect representations learned from classifying cats to be more useful for classifying dogs than for classifying, *e.g.*, foods.

To validate this chain of thought, and more generally, to evaluate the concept generalization capability of representations learned on ImageNet-1K, one key prospect is to understand whether models simply memorize the training data of ImageNet-1K or learn representations that can indeed transfer to unseen concepts. For this, it is important to consider carefully designed experimental protocols where training *vs.* evaluation tasks allow measuring the transferability of representations reliably. Therefore, a principled approach should satisfy the following requirements:

- A set of unseen concepts to evaluate the transferability of representations. Unseen concepts should come from the same concept ontology of the 1000 concepts of ImageNet-1K (which are already regarded as seen concepts), but should be disjoint from them.
- A semantic similarity measure defined between any two concepts. This can be based on, for instance, manually-defined relationship graphs by experts [Miller 1995] or language models representing concepts in a semantic latent space based on textual description of concepts [Mikolov et al. 2013a,b].
- A structured concept space, where the semantic similarity between the seen and unseen concepts are known, for instance, based on the metric defined previously. This would allow examining how the semantic relationships between concepts affect the transfer performance across them.

To our knowledge, there is no benchmark that fulfills all these requirements, and we aim to fill this gap in Chapter 3.

Improving the generalization of supervised learning models

Whether or not image labels are exploited while training models on ImageNet-1K impacts the utility of representations learned by neural networks. They determine to what extent learned representations will be useful for ImageNet-1K or transferable to other tasks.

In the spectrum of utilizing annotations, there are two extremes for ImageNet-1K training. On one hand, there are supervised learning models, which are trained by predicting the image labels annotated by humans. As the task is to distinguish images of the concepts that exist in the training data, learned representations become tailored for those concepts, which are not necessarily useful for images of other (possibly unseen) concepts [Kornblith et al. 2021]. Prior work considered improving the transferability of representations learned by "vanilla" supervised models, which are trained naively by predicting the labels of images with standard architectures, by using alternative loss formulations or modifying the model architecture. For instance, contrastive learning has been used to minimize inter-class similarity of representations while maximizing their intra-class similarity computer either over a large set of samples [Khosla et al. 2020] or small local neighborhoods [Feng et al. 2022]. Orthogonal to the loss formulations, the model architecture has been modified to reduce the overfitting of representations to seen concepts [Wang et al. 2022b]. Although better transfer learning performance has been reported in these works, the resulting models fall behind the state-of-the-art supervised models [Wightman et al. 2021] in terms of their classification accuracy on ImageNet-1K.

On the other hand, there are self-supervised learning models, a branch of unsupervised methods, which learn representations by solving *proxy* tasks depending solely on the images. The idea is to capture visual priors from images that are potentially useful for a wide range of downstream tasks. For this reason, significant research efforts have been devoted to designing proxy tasks that would produce such representations, with the main focus on discriminative tasks [Zhou et al. 2022] rather than generative ones [Brock et al. 2019]. Early works in this direction relied more on low-level tasks such as predicting the orientation of an image [Gidaris et al. 2018] or the color of a gray-scale pixel [Zhang et al. 2016]. More prominent progress has been made with higher-level tasks based on clustering [Caron et al. 2018] or instance discrimination [Dosovitskiy et al. 2016, Wu et al. 2018]. As a result, recent self-supervised approaches have reported better transfer learning performance than their supervised counterparts on various downstream tasks [Caron et al. 2021]. Yet, these models are inferior to the state of the art for ImageNet-1K classification.

As discussed, excelling at both training and transfer tasks is an open challenge, and we rather see a trade-off between these two goals. In Chapter 4, we investigate this trade-off by comparing the performance of models on ImageNet-1K and a number of transfer datasets.

Learning transferable representations from synthetic ImageNet clones

Another important factor that has a crucial impact on the utility of learned representations is training data [Mahajan et al. 2018, Radford et al. 2021a]. More precisely, the quantity and diversity of images or whether they are curated or not (*i.e.*, inspected by humans to meet

certain quality requirements) determine the performance of models. For instance, being a large-scale curated dataset with a fine-grained set of concepts is one of the reasons that makes ImageNet-1K a popular dataset for training models.

More training data generally helps achieving better performance, but it is proportionally more expensive to collect, filter and annotate new data, especially for a dataset like ImageNet-1K [Deng et al. 2009]. How can we expand the training data for ImageNet-1K without going through the trouble of manually collecting new images?

Generating synthetic images to be used as training data is a promising approach in this regard [Besnier et al. 2020]. We can train an image generative model, and task it to produce synthetic images for the 1000 classes of ImageNet-1K. Then, these synthetic images can be used to learn representations. But this approach comes with its own challenges. To start with, how to train the generative model (*i.e.*, by using which data, algorithm or network architecture) is a concern. Synthetic images produced by the generative model should be as realistic as possible, so that the learned representations are useful for the real images of ImageNet-1K. Otherwise, representations would suffer from a potential domain gap between real and synthetic images [Sankaranarayanan et al. 2018, Yang and Soatto 2020]. Training class-conditional generative models is an option in this regard [Brock et al. 2019], but, the quality of synthetic images is often not perfect, with a number of "issues". First of all, the visual fidelity of generated images is often not perfect, *e.g.*, it is quite common to see artifacts in synthetic images such as an image of a duck with no head, or a bird with no wings [Esser et al. 2021]. Moreover, the diversity of generated images, one of the key properties of "good" synthetic images [Baradad Jurjo et al. 2021], is often limited, *e.g.*, multiple random images generated for the same concept may appear in similar pose, background or lighting conditions [Dhariwal and Nichol 2021]. All these issues may limit the generalization capabilities of the representations learned by synthetic images.

In Chapter 5, we investigate ways to produce synthetic ImageNet-1K clones for the purpose of learning transferable image representations.

1.2 Contributions

In this thesis, we tackle the following three research questions to tackle the challenges discussed above:

1. Given a model trained on ImageNet-1K, how can we reliably evaluate its concept generalization capability?
2. How can we learn robust image representations from ImageNet-1K that are not only useful for ImageNet-1K, but also generalize to a wide-range of tasks?

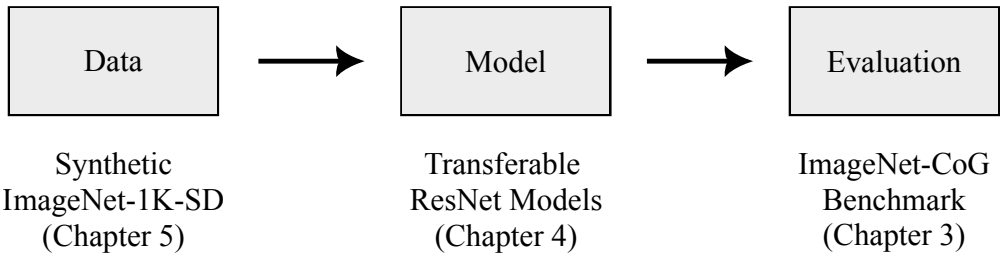


Figure 1.3: Schematic illustration of the three contributions presented in this thesis. (**a**) *Evaluation aspect:* In Chapter 3, we propose a new benchmark (ImageNet-CoG) tailored for evaluating the concept generalization performance of the models trained on ImageNet-1K. (**b**) *Modeling aspect:* In Chapter 4, we propose an improved training setup for training supervised models on the real images of ImageNet-1K. (**c**) *Data aspect:* In Chapter 5, we generate synthetic clones of ImageNet-1K to train supervised models. Note that the illustration is only for a high-level overview, and it omits many technical details.

3. Can we learn such robust image representations from synthetic images generated for the concepts in ImageNet-1K?

This manuscript presents three contributions, one for each research question. Fig. 1.3 gives an illustration of these contributions to aid the reader in positioning them in the context of image representation learning with specific focus on transfer learning.

1. **Measuring concept generalization in visual representation learning** (Chapter 3)
   At the beginning of this PhD program, we observed the remarkable effectiveness of self-supervised models such as MoCo [He et al. 2020] and SimCLR [Chen et al. 2020a] for visual representation learning. Their representations learned on ImageNet-1K could be transferred to downstream image classification, object detection or semantic segmentation tasks on other datasets including Pascal-VOC [Everingham et al. 2009], MS-COCO [Lin et al. 2014], SUN [Xiao et al. 2010], Aircraft [Maji et al. 2013] and others. We sought to determine if the enhanced performance of these models was due to their superior ability to learn representations for the concepts in ImageNet-1K or their increased capacity to generalize effectively to previously unseen concepts. To test this hypothesis, we wanted to understand how much ImageNet-1K "overlaps" with the datasets commonly used for downstream tasks. More specifically, we wanted to check if the concepts in downstream datasets were already seen during training on ImageNet-1K, and if not, how similar they are to the ones in ImageNet-1K in terms of their semantic similarity. It was not straightforward to quantify the overlap between the concepts in ImageNet-1K and those in downstream datasets. This was mainly due to the mismatch between the concept ontologies of all datasets, which were not necessarily aligned. To address this issue, we develop the ImageNet-CoG benchmark

tailored for measuring the concept generalization capability of visual representations learned on ImageNet-1K. In this benchmark, there are five sets of truly unseen concepts (called CoG "levels", *i.e.*, L_1, L_2, ..., L_5), which are sampled from the full ImageNet dataset. Moreover, from the first to the last level, each level contains concepts that are semantically less and less similar to the concepts in ImageNet-1K. We evaluate 31 recent representation learning models including self-supervised models, supervised regularization techniques, model architectures, and make several interesting observations regarding these models.

- This work was published at IEEE International Conference on Computer Vision (ICCV) in 2021 [Sariyildiz et al. 2021].

2. **Improving the generalization of supervised learning models** (Chapter 4)
Our evaluations on ImageNet-CoG reveal that self-supervised models (*e.g.*, DINO [Caron et al. 2021]) are more resilient to concept shift compared to the supervised models which are trained with label-based regularization techniques (*e.g.*, MixUp [Zhang et al. 2018] or CutMix [Yun et al. 2019]) to achieve better performance on ImageNet-1K. A similar trade-off between the ImageNet-1K and transfer performance for supervised models is observed by Kornblith et al. [2021], who show that the best transfer performance is achieved by models which lead to the worst ImageNet-1K performance. These observations indicate that utilizing labels during training a model actually hurts its generalization performance. We argue that this is counter-intuitive and that labels should only help for learning better representations. With this motivation, we tackle the trade-off between ImageNet-1K and transfer performance for the sake of supervised learning to improve their generalization capability while retaining their superior performance on ImageNet-1K. To this end, we design an improved training setup for supervised learning, which includes training components from the best self-supervised models. By training supervised models with our setup, we achieve better transfer performance on 15 image classification tasks (including both large-scale and fine-grained categorization) compared to the state-of-the-art self- and semi-supervised models, DINO [Caron et al. 2021] and PAWS [Assran et al. 2021], respectively. Among the hundreds of models we trained with our setup, we single out two of them: **t-ReX** and **t-ReX*** which are the ResNet50 models with the best transfer and ImageNet-1K performance, respectively.

- The work presented in this chapter was accepted to International Conference on Learning Representations (ICLR) in 2023 [Sariyildiz et al. 2023b].

3. **Learning transferable representations from synthetic ImageNet clones** (Chapter 5)
During 2022, many text-to-image generative models were proposed, with incredible image generation capabilities. We investigated whether these models could be used

for generating images for a dataset like ImageNet-1K to overcome the limited data issue. The main challenge here is that ImageNet-1K is a particular dataset with a large number of fine-grained classes, *e.g.*, many dog breeds, mushrooms types, *etc.*, while these text-to-image models are trained on (image, text) pairs gathered arbitrarily from the internet [Schuhmann et al. 2022]. To this end, we use Stable Diffusion [Rombach et al. 2022] to generate synthetic images for the concepts in ImageNet-1K, and train supervised models on the generated images. To generate images for each class, we need a textual prompt for it, to be given as input to Stable Diffusion. We first examine prompts as simple as only the class name. Upon manual inspection, we encounter a number of issues with such prompts, including images with incorrect semantics or domain, or limited diversity. To address these issues, we considered other class-agnostic prompt alternatives by: **a)** appending the name of the parent class or the description of the class to the class name, **b)** devising prompts where a class is places in one of the backgrounds from the Places365 dataset [Zhou et al. 2017], **c)** reducing the reliance on the textual prompt for Stable Diffusion. These alternative prompts are able to mitigate the issues with simple prompts. We test the performance of the models trained on the generated images produced with each variant of the prompts we investigate, on 5 ImageNet datasets and 15 transfer datasets. We observe that although we obtain lower performance on the ImageNet datasets compared to the baseline models trained on real images, the models trained on the generated images achieve much higher transfer performance.

- The work presented in this chapter was accepted to IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2023 [Sariyildiz et al. 2023a].

1.3 Publications

- "*Concept Generalization in Visual Representation Learning*", by Mert Bulent Sariyildiz, Yannis Kalantidis, Diane Larlus and Karteek Alahari in the IEEE International Conference on Computer Vision (ICCV) 2021 [Sariyildiz et al. 2021]. Code for evaluating models on our benchmark is publicly available on our project website: https://europe.naverlabs.com/research/cog-benchmark
- "*No Reason for No Supervision: Improved Generalization in Supervised Models*", by Mert Bulent Sariyildiz, Yannis Kalantidis, Karteek Alahari and Diane Larlus in the International Conference on Learning Representations (ICLR) in 2023 [Sariyildiz et al. 2023b]. Code for training supervised models with our improved setup and evaluating models on all the transfer datasets is publicly available on our project website: https://europe.naverlabs.com/t-rex

- "*Fake it till you make it: Learning transferable representations from synthetic ImageNet clones*", by Mert Bulent Sariyildiz, Karteek Alahari, Diane Larlus and Yannis Kalantidis in the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2023 [Sariyildiz et al. 2023a]. Our models pretrained on the synthetic ImageNet clones are available on our project website:
  https://europe.naverlabs.com/imagenet-sd

Chapter 2

Related work

Contents

2.1 Image representations 13

2.1.1 Hand-crafted representations 14

2.1.2 Representations learned with neural networks 16

2.2 Forms of supervision to learn representations 19

2.2.1 Supervised learning 20

2.2.2 Self-supervised learning 23

2.3 Learning representations from synthetic data 24

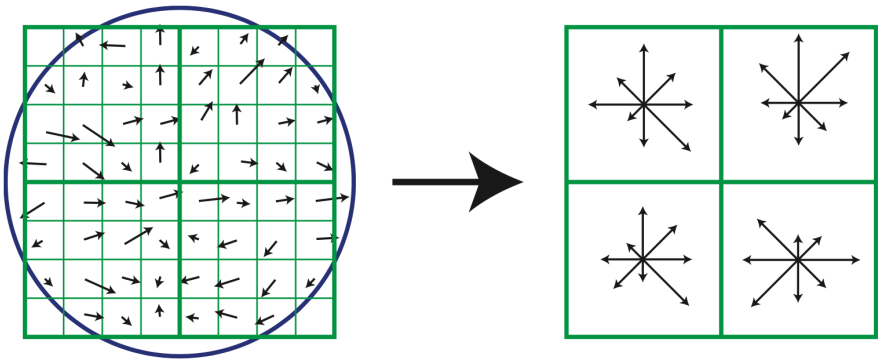
2.4 Evaluating the generalization of representations 25

2.5 Positioning the contributions 28

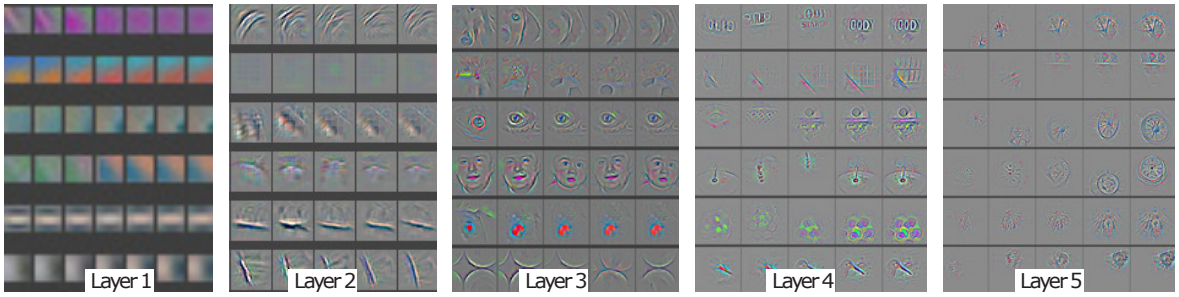
In this chapter, we present previous works related to the research questions presented in Sec. 1.2. We start Sec. 2.1 with a discussion on visual representations, and methods which obtain them using hand-crafted techniques or neural networks. Then, we briefly present different families of approaches to learn "good" visual representations. These include models trained with different forms of supervision (Sec. 2.2), models trained with synthetic data (Sec. 2.3). In Sec. 2.4, we look at ways to evaluate how good visual representations are for different computer vision problems, with emphasis on concept generalization. Finally, we position the contributions presented in the next chapters with respect to these related works in Sec. 2.5.

2.1 Image representations

Extracting image representations (also known as features) is one of the fundamental tasks in computer vision [Bengio et al. 2013]. As image representations facilitate the model designed for the task we want to solve, the "quality" of representations directly affects the performance of the model. Early works relied on human-designed techniques for extracting image features. In the past decade, with deep learning models becoming more and more successful, the field has shifted to *learning* image representations from data tailored for the task itself. This thesis focuses on the latter, *i.e.*, learning image representations by neural



(a) Illustration of pixel gradients (*left*) and SIFT keypoint descriptor based on the histogram of gradients (*right*). Figure taken from [Lowe 2004].



(b) Visualization of hierarchical image features learned by a Convolutional Neural Network. Figure taken from [Zeiler and Fergus 2014].

Figure 2.1: Illustration of image features extracted by (**a**) handcrafted techniques and (**b**) deep neural networks.

networks. Yet, in order to present a complete picture, we briefly discuss the most relevant works in both these paradigms, in the following.

2.1.1 Hand-crafted representations

This refers to features extracted by human-designed rules which exploit low-level image properties, such as pixel colors or gradients. They were primarily used in early 2000s, but then they became gradually obsolete as the community started using neural networks more (which accelerated after the introduction of AlexNet [Krizhevsky et al. 2012]). Here, we present an oversimplified pipeline of extracting such hand-crafted features using a popular method called *bag of visual words*. More comprehensive discussions, including other feature extraction techniques such as Fisher vectors [Perronnin and Dance 2007], can be found in Cinbis [2014], Mensink [2012].

Bag of visual words (BoV) is one of the classical approaches for extracting image features for vision tasks, including *e.g.*, image retrieval [Sivic and Zisserman 2003] or classification [Csurka et al. 2004]. As nicely stated in [Cinbis 2014] "*the main idea is to obtain visual*

words by quantizing local descriptors of image patches (i.e., image regions) with respect to a visual vocabulary", and the whole process involves several important steps outlined as follows:

- Sampling image patches. These patches can be representative image regions (also called as *interest points*) detected based on low-level cues such as edge, corners or blobs [Lindeberg 1998]. Examples of such detectors include SIFT detector [Lowe 2004] or Harris-affine detector [Mikolajczyk and Schmid 2004]. Alternatively, patches can also be sampled densely on a regular grid at multiple scales [Chatfield et al. 2011, Nowak et al. 2006], which is shown to be better for recognition tasks [Nowak et al. 2006].
- Extracting *descriptors* (features) from patches. Once we sample image patches, we extract descriptors from them. The most frequently used descriptors are usually based on gradient orientation histograms such as SIFT [Lowe 2004], SURF [Bay et al. 2008] or DAISY [Winder et al. 2009] descriptors, or color statistics [Perronnin et al. 2010, Van De Weijer and Schmid 2006]. Illustration of the SIFT descriptor is shown in Fig. 2.1a.
- Creating a *visual codebook* and encoding local descriptors via the codebook. A codebook is often constructed by clustering a large set of local descriptors extracted from patches using a clustering algorithm such k-Means. Then each local descriptor is encoded via the codebook, usually by hard or soft cluster assignment.
- Aggregating codes. After encoding each local descriptor by the codebook, a histogram of codes is computed using descriptors for the same image, and they are aggregated by, *e.g.*, average or max pooling.

The BoV approach has limitations. For instance, modeling spatial relationships across patches might be needed for tasks where global layout of objects is important. In those cases, BoV can be modified to incorporate spatial information [Lazebnik et al. 2006], or global image descriptors, such as GIST [Oliva and Torralba 2001] or HOG [Dalal and Triggs 2005] can be extracted. More importantly, as codebook construction and encoding of descriptors are not trivial [Van Gemert et al. 2010], a number of issues arise, ranging from the loss of information due to quantization or ambiguity due to clustering, which has been partially addressed by, *e.g.*, [Avrithis and Kalantidis 2012, Boiman et al. 2008, Jégou et al. 2010, Jurie and Triggs 2005, Philbin et al. 2008, Tuytelaars and Schmid 2007, Van Gemert et al. 2009]. On the other hand, neural networks provide an alternative *data-driven* pipeline for extracting features learned specifically for the task.

2.1.2 Representations learned with neural networks

Brief history of neural networks. The idea behind artificial neural networks dates back to mid 1900s, *e.g.*, McCulloch and Pitts [1943], Rosenblatt [1958], and research in this field has undergone periods of advancement and setback until early 2000s [Goodfellow et al. 2016]. Some notable works in this period include the perceptron model of Rosenblatt [1958], its extension to multiple layers (*i.e.*, multi-layer perceptrons, MLPs) by Ivakhnenko [1971], using backpropagation [Rumelhart et al. 1986, Werbos 1974] for training of neural networks with multiple hidden units, and the introduction of convolutional neural networks (CNN) [Fukushima 1980, LeCun et al. 1989, 1998] to improve the generalization of networks by exploiting certain data biases such as local structures in images, and invariance to translation. (See Goodfellow et al. [2016] for a more detailed discussion.)

Meanwhile, there had also been important developments from the hardware and data perspectives as well. Graphical processing units had been repurposed from gaming to generic massively parallel processing units for neural networks. Large-scale datasets, such as ImageNet [Deng et al. 2009], had been collected for training and evaluating computer vision methods. This led to the pivotal moment of AlexNet [Krizhevsky et al. 2012] winning the ImageNet Large Scale Visual Recognition Challenge (also known as the ImageNet-1K dataset) [Russakovsky et al. 2015] in 2012, which accelerated the adoption of neural networks for large-scale *end-to-end* learning. Subsequently, a multitude of network architectures have been introduced with the aim of enhancing either the computational efficiency or the performance of AlexNet, including ZFNet [Zeiler and Fergus 2014], VGG [Simonyan and Zisserman 2015] Inception [Szegedy et al. 2015] ResNet [He et al. 2016], ResNeXt [Xie et al. 2017], SENet [Hu et al. 2018], DenseNet [Huang et al. 2017], EfficientNet [Tan and Le 2019] ConvNeXt Liu et al. [2022]. Lately, equipped with a self-attention mechanism [Vaswani et al. 2017], Vision Transformers (ViT) [Dosovitskiy et al. 2021] have gained popularity [Liu et al. 2021].

An important qualification of deep neural networks (with many layers) is that they are able to learn hierarchical representations of data [Zeiler and Fergus 2014], tailored for the task they are trained on. Layers in a network encode representations of varying levels of abstraction, depending on their depth in the network. Such representations typically range from low-level features, such as edges, corners and lines, to high-level concepts, such as objects and scenes (an illustration is provided in Fig. 2.1b).

Regularization in neural network training. To achieve great performance with neural networks, they are often trained with certain techniques which either help them better optimize the loss function or improve their generalization as in the traditional machine learning sense [Bishop and Nasrabadi 2006]. Those techniques include, but are not limited to:

- Manipulating the network's weights (parameters) or activations during training. For instance, **a)** injecting noise into the network's activations (*e.g.*, by activation [Srivastava et al. 2014] or layer removal [Huang et al. 2016]), **b)** normalizing activations (*e.g.*, over the samples in the same batch [Ioffe and Szegedy 2015] or the whole activation vector for a single sample [Ba et al. 2016]), **c)** penalizing large weights (*e.g.*, via the weight decay term [Krogh and Hertz 1991] added to the loss function), or **d)** re-parameterizing the weights (*e.g.*, via weight normalization [Salimans and Kingma 2016]).
- Using advanced optimization techniques (*e.g.*, stochastic gradient descent with Adam optimizer [Kingma and Ba 2014, Loshchilov and Hutter 2019] or dynamically adjusting optimizer parameters with schedules [Goyal et al. 2017, Loshchilov and Hutter 2017, Smith et al. 2018]).
- Artificially increasing the amount of effective training data through the use of random image transformations, which is known as data augmentation [Howard 2013, Krizhevsky et al. 2012, Szegedy et al. 2015, Wightman et al. 2021].

The idea of data augmentation is that given an image, several *views* of the image are generated by random data augmentation operations, and the network's output should be similar for these views belonging to the same image. This way, the network learns robust representations invariant to augmentations (similar to the notion of learning view-invariant representations by brains [Den Ouden et al. 2012, Mnih and Kavukcuoglu 2013, Smith and Gasser 2005]), assuming that learning those invariances are helpful for the task [Xiao et al. 2021].

We organize the most commonly used augmentation operations into two groups:

- Label-preserving transformations. These include **a)** altering brightness, contrast, saturation and hue values of an image [Howard 2013], **b)** transforming the color space of an image from RGB to gray-scale or Lab [Tian et al. 2020a], or dropping its color channels [Chen et al. 2020a], **c)** applying filters, *e.g.*, Gaussian Blur or noise, or Sobel filters [Caron et al. 2018], **d)** removing a part of an image, *e.g.*, CutOut [DeVries and Taylor 2017], **e)** spatial and geometric transformations such as horizontal flipping [Krizhevsky et al. 2012], random cropping [Szegedy et al. 2015], translation, rotation, sheering or perspective transform. There are several software packages supporting exhaustive lists of such augmentations, *e.g.*, Albumentations [Buslaev et al. 2020].
- Semantic transformations which change the semantic content on the augmented image, hence its label. Examples include Mix-Up [Zhang et al. 2018] and CutMix [Yun et al. 2019]. Although initially designed for image classification, these augmentations can also be adapted for other tasks such as metric learning [Venkataramanan et al. 2021]

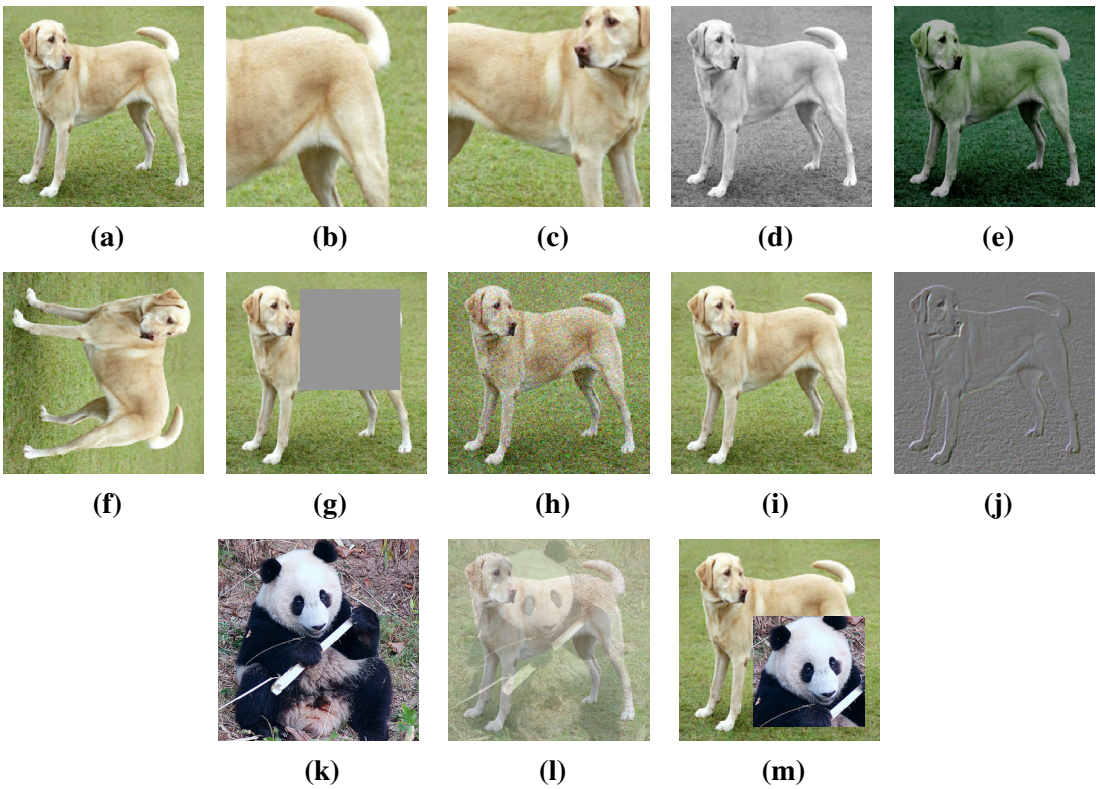


Figure 2.2: Illustrations of the various data augmentation operations. Given an example image in (**a**), the label-preserving augmentations from (**b**) to (**j**) are: (**b**) Crop + resize, (**c**) Crop + resize + flip, (**d**) Color distortion (channel drop), (**e**) Color distortion (jitter), (**f**) Rotation, (**g**) Cutout, (**h**) Gaussian noise, (**i**) Gaussian blur and (**j**) Sobel filtering. Provided another image in (**k**), semantic transformations of (**a**) and (**k**) are (**l**) MixUp and (**m**) CutMix. Each augmentation operation has one or more parameters determining the output image. The images in (**a**) to (**j**) are taken from [Chen et al. 2020a], and the image in (**k**) is from ImageNet [Russakovsky et al. 2015].

or self-supervised learning [Shen et al. 2022]. Moreover, such semantic transformations can be applied in the space of learned representations [Kalantidis et al. 2020, Venkataramanan et al. 2022, Verma et al. 2019], rather than input images.

An illustration of some of those data augmentation operations is shown in Fig. 2.2.

Transferring representations to other tasks. Researchers found that robust representations learned on one task can also generalize to, *i.e.*, *transfer* to, other related vision tasks [Girshick et al. 2014, Razavian et al. 2014, Yosinski et al. 2014]. This means, for instance, a network can be *pretrained* for image classification on ImageNet-1K, and representations learned by this network can be re-used on another task, *e.g.*, semantic segmentation or object detection on the MS-COCO dataset [Lin et al. 2014]. This paradigm of transferring representations from one task (*i.e.*, training task) to another one (*i.e.*, transfer task) is often denoted as transfer

learning [Bozinovski 2020, Bozinovski and Fulgosi 1976], and constitutes the main focus of this thesis.

As there are many hidden layers in a deep network, the choice of which layer to extract representations from is not straightforward. Using features from intermediate layers of networks has been considered before, *e.g.* for training object detectors [Lin et al. 2017] and image classification models [Lee et al. 2015], or evaluating the transferability of individual layers [Gidaris et al. 2018, Zhang et al. 2016] or groups of layers [Evci et al. 2022]. However, selecting optimal layers for each problem is infeasible due to the computational nature of this selection. Therefore, recent approaches measuring transferability of representations [Caron et al. 2021, Chen et al. 2020a, He et al. 2020] often extract features from a single layer, which is the penultimate layer of the network. More concretely, consider a model to be composed of an encoder f_θ and a task head g_ϕ, parameterized by θ and ϕ, respectively, *i.e.*, model outputs $o = g_\phi(f_\theta(I))$:

- $f_\theta : I \rightarrow x \in \mathbb{R}^d$ is an encoder (also known as "backbone") mapping images I to d-dimensional representations x. These are the representations transferred to other tasks or datasets. For instance, a common practice is to train a linear classifier, *e.g.*, support vector machines (SVM) classifier [Vapnik 1999], on top of representations x extracted for the transfer task. We discuss this in more detail in Sec. 2.4.
- $g_\phi : x \rightarrow o \in \mathbb{R}^{|O|}$ is a module mapping representations x to outputs of the model, *i.e.*, predictions according to the task being considered.

For instance, in the case of training an image classification model, f_θ could be any recent deep network mentioned earlier and g_ϕ could be *class weights* $W \in \mathbb{R}^{|C| \times d}$ mapping representations to class prediction scores $o = Wx$, where $|C|$ is the number of classes.

In the next section, we look at some of the prominent ways of pretraining neural networks to learn robust representations that can transfer.

2.2 Forms of supervision to learn representations

Previous section discussed two distinct mechanisms for representing images, *i.e.*, hand-crafted features *vs.* features learned by neural networks. This thesis focuses on the latter, and in particular, on the way neural networks are trained to learn representations. In fact, depending on the constraints of the problem at hand, there are several ways to learn visual representations. For instance, on one hand, supervised learning is applicable when human-provided annotations are given for images [Kornblith et al. 2021]. On the other hand, unsupervised or self-supervised learning [Caron 2021] is an alternative paradigm to learn strong visual priors when no annotation is available. As we move along the annotation spectrum in

between those two paradigms, there are also semi-supervised [Assran et al. 2021, Siméoni et al. 2021] and weakly-supervised [Desai and Johnson 2021, Mahajan et al. 2018, Radford et al. 2021a, Sariyildiz et al. 2020b] approaches. There are other paradigms such as knowledge distillation [Buciluǎ et al. 2006, Hinton et al. 2014], which provides tools to transfer knowledge from one model (*teacher*) to another one (*student*) [Budnik and Avrithis 2021].

In the remainder of this section, we will briefly discuss related works in supervised (Sec. 2.2.1) and self-supervised (Sec. 2.2.2) learning. Each methodology has its own advantages and disadvantages. For instance, supervised learning is the de facto approach to learn the most useful representations for a given task, *e.g.*, image classification. Yet, the transferability of learned representations is largely affected by the level of regularization applied during training [Kornblith et al. 2019, 2021]. Self-supervised learning, on the other hand, has been shown to learn more transferable representations than supervised learning [Caron 2021].

2.2.1 Supervised learning

In this setting, given an image I, a model is trained by predicting ground-truth annotations y of the image. Throughout this thesis, we focus on the case where such annotations are in the form of image labels, *i.e.*, the category label of an object in an image denoted by $y \in \{0,1\}^{|C|}$, a $|C|$-dimensional one-hot label vector, where $|C|$ is the number of classes. Following the notation we introduced earlier, we define the components of an image classification model g_ϕ as follows:

- $f_\theta : I \to x \in \mathbb{R}^d$ is an encoder producing representations x of an image I.
- $g_\phi : x \to o \in \mathbb{R}^{|C|}$ is a module for predicting class labels. For simplicity, we assume that $g_\phi(x) = Wx + b$, where $W = \{w_c \in \mathbb{R}^d\}_{c=1}^C$ and $b \in \mathbb{R}^C$ are class weights and bias terms, respectively. They can be both trainable or non-trainable, *i.e.*, not updated after initialization in the latter case [Hoffer et al. 2018, Sariyildiz et al. 2020a].

Several loss functions have been used to train the parameters of this image classification model (*i.e.*, θ, W and b).

Multi-class cross-entropy (MCCE) (also known as softmax cross-entropy) is a standard loss function used for multi-class classification problems. When training models with this function, we compute class scores by multiplying image representations $x = f_\theta(I)$ with class weights W and adding bias terms b. Then, these class scores are turned into class probabilities using softmax, to finally compute log loss:

$$\mathcal{L}_{\text{MCCE}}(x,y) = -\sum_{c=1}^{C} y_{[c]} \log \frac{\exp(x^\top w_c + b_c)}{\sum_{k=1}^{C} \exp(x^\top w_k + b_k)}. \tag{2.1}$$

Binary cross-entropy (BCE) is mainly used for binary classification problems. Yet, by following a one-versus-rest strategy, it can used in multi-class classification problems as well [Beyer et al. 2020, Wightman et al. 2021]:

$$\mathcal{L}_{\text{BCE}}(x,y) = -\sum_{c=1}^{C}\left[y_{[c]}\log\left(\frac{\exp(x^\top w_c + b_c)}{\exp(x^\top w_c + b_c)+1}\right) + (1-y_{[c]})\log\left(1-\frac{\exp(x^\top w_c + b_c)}{\exp(x^\top w_c + b_c)+1}\right)\right], \tag{2.2}$$

Different from MCCE, prediction probability for class c_i does not affect that of class c_j for $i \neq j$, as we only use w_c to make a prediction for the class c.

Cosine-softmax cross-entropy (CSCE) [Kornblith et al. 2021] is a variant of MCCE where both representations x and class weights w_c (for all c), are ℓ_2-normalized:

$$\mathcal{L}_{\text{CSCE}}(x,y) = -\sum_{c=1}^{C} y_{[c]}\log\frac{\exp(\bar{x}^\top \bar{w}_c/\tau + b_c)}{\sum_{k=1}^{C}\exp(\bar{x}^\top \bar{w}_k/\tau + b_c)}, \tag{2.3}$$

where $\bar{x} = \frac{x}{\|x\|}$, $\bar{w_c} = \frac{w_c}{\|w_c\|}$ and τ is a temperature parameter controlling the smoothness of the probability distribution.

In the cross-entropy variants discussed above, bias terms b can often be discarded when training models on datasets with *balanced* data, *i.e.*, datasets where the number of images per class is roughly the same, such as ImageNet-1K. This is especially the case for the CSCE variant, as bias terms can artificially alter the effect of the temperature parameter τ, which is an important hyper-parameter to be carefully set.

Nearest class mean classifier (NCM) [Mensink et al. 2013] modifies the cross-entropy formulation above by replacing class weights W by *class prototypes* $U = \{u_c \in \mathbb{R}^d\}_{c=1}^{C}$, where $u_c = \frac{1}{|X_c|}\sum_{x\in X_c}$ is the prototype of class c obtained by averaging features of the images that belong to class c (denoted by X_c), and $|.|$ denotes the cardinality of a set. Then, an image is assigned to the class with the closest mean according to a distance measure, which can simply be Euclidean distance $d(x,u) = \|(x-u)\|$ or a learned Mahalanobis distance $d(x,u) = (x-u)^\top M(x-u)$ with parameters M. In this formulation, computing prototypes requires processing all the images in the dataset, which is not feasible for large datasets. But this is not a problem if image features are static, *e.g.*, obtained by the techniques discussed above in Sec. 2.1.1, as prototypes can be computed once in the beginning of training. However, adapting this formulation in the context of deep learning, where representations are updated after each training iteration, is not trivial, *i.e.*, as with representations, class prototypes must also be updated. Guerriero et al. [2018] proposed to approximate class prototypes

by updating them in an online manner, using samples in the batch at each step of training. This approximation circumvents the need of processing all the images in the dataset at each training step, but it is not as accurate as the original NCM formulation.

Supervised contrastive learning (SCL), proposed in SupCon [Khosla et al. 2020], is different from the other loss functions mentioned above in a way that it does not utilize class weights or prototypes to predict the label of an image. Instead, it is a contrastive learning approach (adapted from a self-supervised model SimCLR [Chen et al. 2020a]), where representations of images from the same class are enforced to be close to each other, while representations from different classes are pushed apart. In their work, Khosla et al. [2020] discuss two ways to extend SimCLR for supervised learning, which are reminiscent of neighborhood component analysis (NCA) of Goldberger et al. [2004]. One of them leads to substantially better results than the other (denoted as $\mathcal{L}_{out}^{sup}$ in [Khosla et al. 2020]), and it is formulated as follows:

$$\mathcal{L}_{\text{SupCon}}(x,y) = -\frac{1}{|P(x,y)|} \sum_{x^+ \in P(x,y)} \log \frac{\exp(\bar{x}^\top \bar{x}^+ / \tau)}{\sum_{x^- \in N(x,y)} \exp(\bar{x}^\top \bar{x}^- / \tau)}, \tag{2.4}$$

where $P(x,y)$ and $N(x,y)$ denote the set of positive and negative pairs for the representation x corresponding to an image with label y. As it is often beneficial to have a large pool of negative pairs in contrastive learning, SupCon circumvents the need for large batches by adding a momentum and a memory bank similar to another self-supervised model MoCo [He et al. 2020].

Following the line of SupCon, there are also other works which have taken inspiration from self-supervised learning. Supervised-MoCo [Zhao et al. 2021b] filters out false negatives in the memory bank of MoCo using image labels, while LOOK [Feng et al. 2022] modifies the NCA objective to only consider the closest neighbors of each query image.

So far, we assumed g_ϕ to be a simple linear layer producing model outputs. However, following the recent practice in self-supervised learning, g_ϕ can include a multi-layer perceptron with several hidden layers. This practice is shown to increase the generalization capability of encoders f_θ [Wang et al. 2022b].

Although the loss functions mentioned above effectively exploit image annotations to learn representations, collecting them is an expensive and error-prone process, especially for fine-grained categorization. Moreover, it imposes certain labeling biases, limiting what can be predicted in natural images, which can possibly contain tens of objects interacting with each other in one way or another [Caron 2021]. The following section discusses methods for visual representation learning in an annotation-free manner.

2.2.2 Self-supervised learning

Self-supervised learning (SSL), a type of unsupervised learning, is a methodology for training models without requiring annotations. In this setting, a training task is still needed to learn representations, but it does not depend on annotations (*e.g.*, image labels provided by humans as in supervised learning we mentioned above). Such training tasks are treated as a *proxy* to learn visual representations, hence, they are often called *proxy tasks*.

There are different families of SSL approaches to learn representations, including

- image generating models, which reconstruct pixels from representations of individual images, such as Autoencoders (either image-level [Hinton and Salakhutdinov 2006] or patch-level [He et al. 2022, Kakogeorgiou et al. 2022]), GANs [Brock et al. 2019] or autoregressive models [Van den Oord et al. 2016],
- view-invariant learning approaches [Chen et al. 2020a, Dosovitskiy et al. 2016, Grill et al. 2020, Hadsell et al. 2006, Misra and van der Maaten 2020, Tian et al. 2020a, Zbontar et al. 2021], which take into account pairwise relations formed by data augmentation,
- clustering approaches, which build on the idea that clusters should be formed in representation spaces, and they can be **a)** predicted by their assignment scores [Asano et al. 2020, Caron et al. 2018, 2020], **b)** locally approximated based on pairwise similarity of samples [Koohpayegani et al. 2021] or **c)** modeled by a probabilistic approach [Li et al. 2021b],
- other pretext tasks from data such as colorization [Zhang et al. 2016], patch prediction [Doersch et al. 2015], frame sequence prediction [Misra et al. 2016], rotation prediction [Gidaris et al. 2018] or prediction of bag-of-words [Gidaris et al. 2021] or HOG [Wei et al. 2022] features.

We refer the reader to Asano [2021], Caron [2021] for a detailed history on self-supervised learning.

Recent SSL models are shown to learn more transferable representations than their supervised counterparts [Grill et al. 2020]. Thanks to this ability, they can also be used to pretrain representations which are then fine-tuned with supervised learning objectives [Bourcier et al. 2022b]. This practice is especially useful in the case of limited labeled data [Bourcier et al. 2022a]. It is worth noting some of the key components of the recent successful SSL approaches:

- Heavy data augmentation pipelines [Chen et al. 2020a], including *e.g.*, creating multiple augmented views of images in the same batch, also known as *multi-crop* augmentation [Caron et al. 2020, Hoffer et al. 2020]. Multi-crop, creates crops with different scales and resolutions, producing challenging views of an image for which the model is encouraged to learn consistent representations [Assran et al. 2021, Caron et al. 2021].
- Keeping a large pool of consistent negatives for better performance in contrastive learning, using, for instance, a large batch size [Chen et al. 2020a] or memory bank [He et al. 2020].
- Including a number of non-linear hidden layers in the task head g_ϕ, also known as *projector head* [Chen et al. 2020a]. The size of projector heads is set carefully depending on the task. In some cases, they can be much bigger than encoders f_θ [Zbontar et al. 2021].
- Long training schedules, *e.g.*, up to 1600 epochs Chen et al. [2020c], He et al. [2022], which usually improves performance.

2.3 Learning representations from synthetic data

Until now, we focused on learning visual representations from real images. Recently, following the advances in generative models (see, *e.g.*, Lucas [2020]), training models with synthetic data has gained popularity. As more and diverse data usually improves performance [Hestness et al. 2017], the image generation process of generative models can be controlled by side information, such as class labels or textual prompts (*e.g.*, see Fig. 2.3), in order to synthesize images for a particular task for training better models. In this section, we look at such recent approaches that train models on synthetic data.

Learning with synthetic data has been considered as a way to create large amounts of labeled data for annotation heavy tasks, such as understanding human motion or their 3D shapes [Guo et al. 2022, Mahmood et al. 2019, Pumarola et al. 2019, Varol et al. 2017], semantic segmentation [Chen et al. 2019b, Li et al. 2021a, 2022, Sankaranarayanan et al. 2018, Tritrong et al. 2021], optical flow estimation [Dosovitskiy et al. 2015, Mayer et al. 2016, whan Kim et al. 2022] or dense visual alignment [Peebles et al. 2022]. In most cases, this synthetic data requires access to 3D models and renderers [Mahmood et al. 2019, Zheng et al. 2020] or to a simulator [Amini et al. 2020, de Melo et al. 2021, Dosovitskiy et al. 2017, Richter et al. 2016] with a physically plausible 3D graphics engine. Kataoka et al. [2022] recently proposed pretraining on a database of synthetic fractal images before fine-tuning the model using real images on a downstream task. Kumar et al. [2022] generates synthetic OCT images to train a glaucoma detection model to be applied to real images.

(a) A cute corgi lives in a house made out of sushi.

(b) A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.

(c) A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.

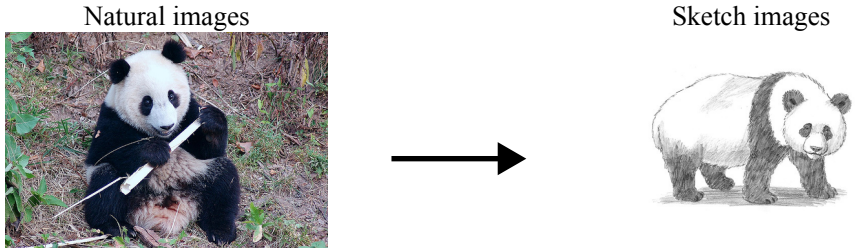
Figure 2.3: Synthetic images generated by Imagen for the given textual prompts. Figure taken from Saharia et al. [2022].

Data sampled from generative models [Goodfellow et al. 2020, Ho et al. 2020, Nichol et al. 2021, Ramesh et al. 2021, Rombach et al. 2022, Saharia et al. 2022] can be seen as data with added functionalities or "data++" [Isola 2022]. Such data can be manipulated, interpolated or composed [Chai et al. 2021a,b, Goetschalckx et al. 2019, Jahanian et al. 2020, 2022] with dedicated operators in their latent space, and further used for counterfactual reasoning [Liu et al. 2019, Mao et al. 2021, Oktay et al. 2018, Sauer and Geiger 2021].

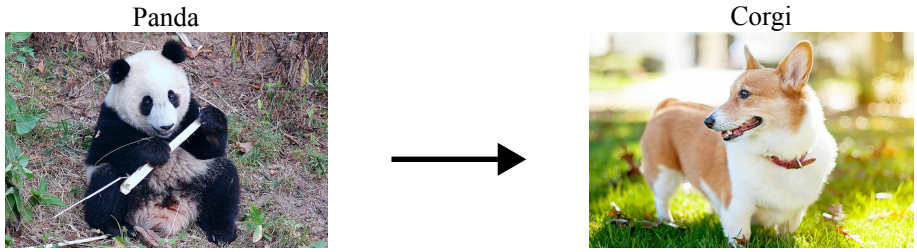
Synthetic ImageNet clones. Synthetic images for ImageNet classes appear in a number of related works [Besnier et al. 2020, Li et al. 2022, Ravuri and Vinyals 2019] using class conditional Generative Adversarial Networks (GANs), such as BigGAN [Brock et al. 2019]. Besnier et al. [2020] generate images for ten ImageNet classes and propose techniques to reduce the gap between models trained on generated images and real ones. Li et al. [2022] synthesize five images for each ImageNet-1K class, together with their semantic segmentation annotations to automatically generate pixel-level labels at scale.

2.4 Evaluating the generalization of representations

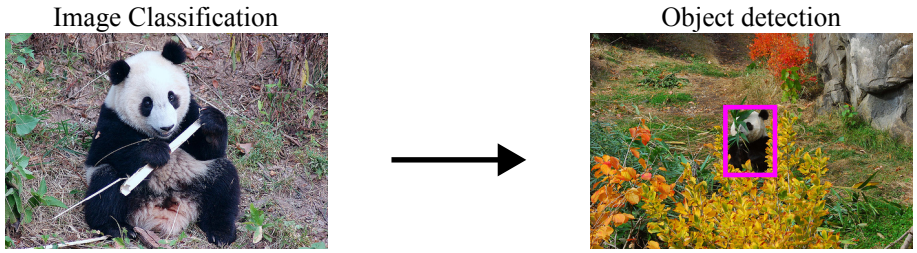
In the previous two section, we discussed works on learning visual representations, *i.e.*, the model training aspect. Yet, it is equally important to comprehend the strengths and limitations of these models, such as whether they merely overfit to the task they train on



(a) Domain generalization from natural to sketch images of a panda.



(b) Concept generalization from the "panda" category to the "corgi" category.



(c) Task generalization from image classification to object detection.

Figure 2.4: Illustration on the several aspects of generalization in computer vision. All panda images are from Russakovsky et al. [2015], except the sketch image of a panda in (**a**), which is from Wang et al. [2019].

or demonstrate reliable generalization to unseen data. This section reviews works on this evaluation aspect.

Generalization. Models are usually evaluated by measuring their generalization capability to unseen data, *i.e.*, how well they perform on unseen data. From machine learning point of view, generalization has been studied under different perspectives such as

- regularizing models by imposing architectural constraints [Larsson et al. 2017, Srivastava et al. 2014] or using data augmentation to artificially increase the amount of data (as discussed earlier),
- finding links to human cognition [Geirhos et al. 2018], or
- developing quantitative metrics to better understand it, *e.g.*, through loss functions [Li et al. 2018] or complexity measures [Neyshabur et al. 2017].

Several dimensions of generalization have also been explored in the context of computer vision, for instance, i) generalization to different visual distributions of the same concepts (domain adaptation) [Csurka 2017], ii) generalization across semantic concepts, which is a crucial part of zero-shot [Lampert et al. 2009, Socher et al. 2013] and few-shot [Vinyals et al. 2016] learning, or iii) generalization across tasks [Zamir et al. 2018]. An illustration highlighting these three aspects is provided in Fig. 2.4. In many of these scenarios, ImageNet-1K [Russakovsky et al. 2015] has played a key role, *i.e.*, models pretrained on ImageNet-1K have usually been better starting points for tackling these problems [Huh et al. 2016].

In the following, we look at common practices for measuring generalization across vision tasks. Then we discuss works on concept generalization.

Evaluations through transfer learning. As we mentioned in Sec. 2.1.2, transferring representations from one task to another one is often denoted as transfer learning [Bozinovski and Fulgosi 1976]. In this context, there are several works studying transfer learning from a theoretical stand point [Baxter 2000, Tripuraneni et al. 2020]. A more common approach is to evaluate representations on a wide-range of *downstream* tasks benchmarking models from different aspects of generalization [Islam et al. 2021, Kolesnikov et al. 2020]. As performing downstream tasks can be a cumbersome endeavor, recent works also compute proxy metrics on representations which are expected to account for the transferability of representations without performing downstream tasks.

Downstream tasks. When it comes to evaluating the quality of visual representations, the gold standard is to benchmark models by solving diverse tasks such as classification, detection, segmentation and retrieval on many datasets [Caron et al. 2019, Chen et al. 2020a, Ericsson et al. 2021, Goyal et al. 2019, He et al. 2020, Kornblith et al. 2019, Zhai et al. 2019b]. Several benchmarks have been proposed as evaluation suites [Goyal et al. 2019, Zhai et al. 2019b]. Besides the training task of ImageNet-1K, the most commonly used datasets include Places [Zhou et al. 2017], Pascal-VOC [Everingham et al. 2009], MS-COCO [Lin et al. 2014]. When benchmarking models, they can either be *fine-tuned* or used as a *feature extractor*. In the former, their parameters are updated for the downstream task, whereas in the latter, their parameters are frozen and only the additional parameters introduced for the downstream task are trained. Therefore, fine-tuning models is often computationally more expensive than using them as feature extractors. On the other hand, an important point when using models as feature extractors is to decide on which layer to extract representations from, which might affect performance [Goyal et al. 2019]. As mentioned earlier (in Sec. 2.1), the selection of the optimal layer, or even, combination of layers is infeasible in practice. To

work around this problem one can use an expendable projector head [Chen et al. 2020a] to obtain representations with desired transferability characteristics.

Transferability scores. Performing downstream tasks is a reliable way of measuring transferability of representations, but this process can be computationally demanding, *e.g.*, when fine-tuning models on large-scale downstream datasets. There have been efforts to sidestep downstream evaluations by computing certain metrics on top of representations, which are easier to obtain compared to performing downstream tasks, with the assumption that such metrics are expected to correlate well with downstream task performance. For instance, instead of, *e.g.*, training an image classifier on a downstream dataset, Bao et al. [2019] measure inter-class variance and feature redundancy of representations extracted for images of the downstream dataset or Pándy et al. [2022] measures the statistical overlap between the classes in a downstream dataset (measured by Gaussian Bhattacharyya coefficient) using their representations. See Agostinelli et al. [2022] for a recent study on the popular transferability metrics.

2.5 Positioning the contributions

We position the contributions of this thesis (presented in Chapters 3 to 5) with respect to the related works discussed in this chapter. Recall that our contributions tackle three research questions (listed in Sec. 1.2 and repeated here for convenience):

1. Given a model trained on ImageNet-1K, how can we reliably evaluate its concept generalization capability?
2. How can we learn robust image representations from ImageNet-1K that are not only useful for ImageNet-1K, but also generalize to a wide-range of tasks?
3. Can we learn such robust image representations from synthetic images generated for the concepts in ImageNet-1K?

A benchmark for measuring concept generalization (Chapter 3). As we noted in Sec. 2.4, the quality of the learned visual representations is usually determined based on downstream tasks covering multiple generalization facets, such as domain adaptation, task transfer or concept generalization. Existing benchmarks [Goyal et al. 2019, Guo et al. 2020] focus more on the first two facets and a more principled analysis is needed for the last one, *i.e.*, the impact of the semantic relationship between the concepts seen during training and those seen during evaluation (seen and unseen concepts, respectively) has been overlooked in evaluating the quality of visual representations. To close this gap, we present a controlled benchmark,

named ImageNet-CoG, that factors in such relations, while keeping all the other generalization facets fixed. Our benchmark is built on the full ImageNet dataset [Deng et al. 2009]: Seen concepts are the 1000 concepts from the ImageNet-1K Russakovsky et al. [2015] dataset (a subset of the full ImageNet) and we select 5000 unseen concepts from the remaining of ImageNet. This way, we make sure that seen and unseen concepts are disjoint. Moreover, those unseen concepts are split into 5 groups (which we call *levels*) such that from the first to last level, each level contains semantically less and less similar concepts with respect to the seen ones. This allows us to evaluate the impact of the semantic relationship between seen and unseen concepts to transferability across them.

Improving the generalization of supervised models (Chapter 4). We mentioned in Sec. 2.2.2 that recent self-supervised learning (SSL) approaches learn more transferable representations than their supervised counterparts. In fact, we verify this observation for concept generalization using our ImageNet-CoG benchmark (see Sec. 3.4). To understand this phenomenon, we train supervised models equipped with some of the key components of the successful SSL approaches (listed in Sec. 2.2.2) using the cosine-softmax cross-entropy (CSCE) loss (Equation (2.3)). We observe that multi-crop augmentation and expendable projector heads boost the generalization performance of supervised models, *i.e.*, prevent their encoders f_θ from overfitting too much to the training task and allow them to learn more transferable representations. Moreover, we replaced class weights in CSCE with class prototypes as in the NCM approach of [Mensink et al. 2013] (classifier described in Sec. 2.2.1). Different from NCM and DeepNCM [Guerriero et al. 2018], we compute class prototypes using representations stored in an online memory bank [He et al. 2020].

Learning transferable representations using synthetic images (Chapter 5). We mentioned that ImageNet-1K is one of the most frequently used dataset to learn visual representations (Secs. 2.1.2 and 2.4), and that several previous works generated synthetic ImageNet-1K clones using conditional generative models (Sec. 2.3). Such ImageNet-1K clones have mostly been produced by *label-conditioned* Generative Adversarial Networks [Goodfellow et al. 2014] trained on ImageNet-1K itself. Yet, recent text-to-image synthesis models are more generic thanks to the large-scale image-text data [Schuhmann et al. 2022] they trained on, and they produce more realistic images thanks to the advances in image-generative modeling [Rombach et al. 2022]. We revisit training models on synthetic ImageNet-1K clones by creating them using Stable Diffusion [Rombach et al. 2022]. Moreover, we evaluate the quality of representations learned from these synthetic images not only for ImageNet datasets, but also for transfer datasets, and the latter leads to interesting observations.

Chapter 3

Measuring concept generalization in visual representation learning

Contents

3.1 Introduction 32

3.2 Related work 35

3.3 The ImageNet-CoG benchmark 36

3.3.1 Seen concepts 37

3.3.2 Selecting eligible unseen concepts 37

3.3.3 Concept generalization levels 38

3.3.4 Evaluation protocol 40

3.4 Evaluating models on ImageNet-CoG 42

3.4.1 Models 42

3.4.2 Results 45

3.5 Further analysis on ImageNet-CoG 50

3.5.1 What if we fine-tuned the backbones? 50

3.5.2 word2vec as an alternative semantic similarity 51

3.5.3 Potential label noise in ImageNet-CoG 54

3.6 Conclusion 54

In this chapter, we present the first contribution of this thesis, which was presented at IEEE International Conference on Computer Vision (ICCV) in 2021 [Sariyildiz et al. 2021]. Here, our focus is on the evaluation aspect of visual representations, *i.e.*, measuring how "useful" visual representations are for transfer tasks. More specifically, we look at this evaluation aspect from the concept generalization perspective for models pretrained on ImageNet-1K.

As we have discussed in Chapter 2, measuring concept generalization, *i.e.*, the extent to which models trained on a set of (seen) visual concepts can be leveraged to recognize a new set of (unseen) concepts, is an important aspect of evaluating visual representations. Nonetheless, the choice of unseen concepts for such an evaluation is usually made arbitrarily,

and independently from the seen concepts used to train representations, thus ignoring any semantic relationships between the two.

To address this problem, we propose **ImageNet-CoG**,[1] a novel benchmark on the ImageNet-21K (IN-21K) dataset that enables measuring concept generalization in a principled way. Utilizing our benchmark, we show that the semantic relationships between seen and unseen concepts indeed affect generalization performance.

Our benchmark leverages expert knowledge that comes from WordNet in order to define a sequence of unseen IN-21K concept sets that are semantically more and more distant from the ImageNet-1K (IN-1K) subset, a ubiquitous training set. This allows us to benchmark visual representations learned on IN-1K out-of-the box. We conduct a large-scale study encompassing 31 convolution and transformer-based models and show how different architectures, levels of supervision, regularization techniques and use of web data impact the concept generalization performance.

3.1 Introduction

There has been an increasing effort to tackle the need for manually-annotated large-scale data in deep models via transfer learning, *i.e.*, by transferring representations learned on resourceful datasets and tasks to problems where annotations are scarce. Prior work has achieved this in various ways, such as, imitating knowledge transfer in low-data regimes [Vinyals et al. 2016], exploiting unlabeled data in a self-[He et al. 2020] or weakly- [Mahajan et al. 2018] supervised manner.

The quality of the learned visual representations for transfer learning is usually determined by checking whether they are useful for, *i.e.*, generalize to, a wide range of downstream vision tasks. Thus, it is imperative to quantify this generalization, which has several facets, such as generalization to different input distributions (*e.g.*, from synthetic images to natural ones), to new tasks (*e.g.*, from image classification to object detection), or to different semantic concepts (*e.g.*, across different object categories or scene labels). Although the first two facets have received much attention recently [Goyal et al. 2019, Guo et al. 2020], we observe that a more principled analysis is needed for the last one.

As also noted by Deselaers and Ferrari [2011], Yosinski et al. [2014], the effectiveness of knowledge transfer between two tasks is closely related to the semantic similarity between the concepts considered in each task. However, assessing this relatedness is not straightforward, as the semantic extent of a concept may depend on the task itself. In practice, models consider an exhaustive list of downstream tasks that cover a wide range of concepts [Chen

[1] https://europe.naverlabs.com/cog-benchmark

et al. 2020a, Kornblith et al. 2019] in order to test their transfer learning capabilities. Previous attempts discussing this issue have been limited to intuition [Yosinski et al. 2014, Zhao et al. 2021b]. We still know little about the impact of the *semantic relationship* between the concepts seen during training visual representations and those seen during their evaluation (*seen* and *unseen* concepts, respectively).

In this chapter, we study the generalization capabilities of visual representations across concepts that exist in a large, popular, and broad ontology, the subset of WordNet [Miller 1995] used to build ImageNet-21K [Deng et al. 2009] (IN-21K), while keeping all the other generalization facets fixed. Starting from a set of seen concepts, the concepts from the popular ImageNet-1K [Russakovsky et al. 2015] (IN-1K) dataset, we leverage semantic similarity metrics based on this ontology crafted by experts to measure the semantic distance between IN-1K and every unseen concept (*i.e.*, any concept from IN-21K that is not in IN-1K). We rank unseen concepts with respect to their distance to IN-1K and define a sequence of five, IN-1K-sized *concept generalization levels*, each consisting of a distinct set of unseen concepts with increasing semantic distance to the seen ones. This results in a large-scale benchmark that consists of five thousand concepts, that we refer to as the **ImageNet Co**ncept **G**eneralization benchmark, or **ImageNet-CoG** in short. The benchmark construction process is illustrated in Fig. 3.1.

Given a model trained on IN-1K, the evaluation protocol for ImageNet-CoG consists of two phases: it first extracts features for images of IN-1K and of the five concept generalization levels, and then learns individual classifiers, for each level, using a varying amount of samples per concept. By defining the set of seen concepts for our benchmark to be IN-1K classes, we are able to evaluate models trained on IN-1K out-of-the box. We therefore use publicly available pretrained models and analyse a large number of popular models under the prism of concept generalization.

Our contributions in this chapter are as follows:

- We propose a systematic way to study concept generalization, by defining a set of seen concepts along with sets of unseen concepts that are semantically more and more distant from the seen ones.
- We design ImageNet-CoG, a large-scale benchmark, which embodies this systematic way. It is designed to evaluate models pretrained on IN-1K out-of-the-box and draws unseen concepts from the rest of the IN-21K dataset. We measure concept generalization performance on five, IN-1K-sized levels, by learning classifiers with a few or all the training images from the unseen concepts.

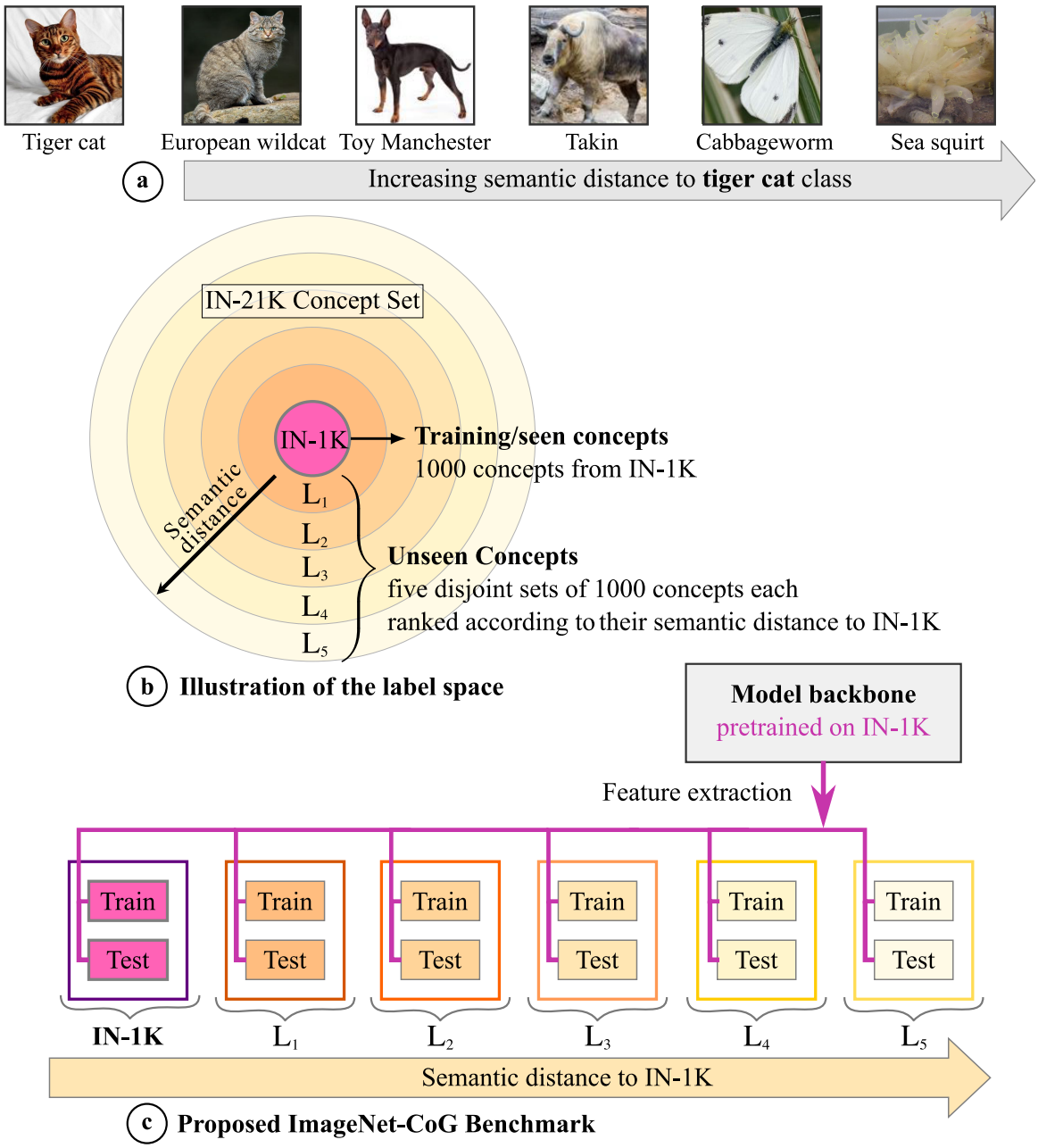


Figure 3.1: An overview of our Concept Generalization (CoG) benchmark. (a) An example of five concepts from the ImageNet-21K dataset [Deng et al. 2009] (IN-21K), ranked by increasing *semantic* distance (decreasing Lin similarity [Lin 1998]) to the ImageNet-1K (IN-1K) dataset [Russakovsky et al. 2015] concept "Tiger cat". (b) We rank the 21K concepts of IN-21K according to their semantic distance to the 1000 concepts of IN-1K and split the ranked list to extract 5 groups of 1000 concepts. We refer to the five IN-1K-sized datasets of increasing semantic distance from IN-1K as *concept generalization levels*, denoted as $L_{1/2/3/4/5}$. (c) The proposed ImageNet-CoG benchmark uses a model trained on IN-1K as a feature extractor and evaluates its concept generalization capabilities by learning linear classifiers for each level of more and more challenging unseen concepts.

- We conduct a large-scale study benchmarking 31 state-of-the-art visual representation learning approaches on ImageNet-CoG and analyse how different architectures, levels of supervision, regularization techniques and additional web data impact the concept generalization performance, uncovering several interesting insights.

3.2 Related work

Generalization has been studied under different perspectives such as regularization [Srivastava et al. 2014] and augmentation [Yun et al. 2019] techniques, links to human cognition [Geirhos et al. 2018], or developing quantitative metrics to better understand it, *e.g.*, through loss functions [Li et al. 2018] or complexity measures [Neyshabur et al. 2017]. Several dimensions of generalization have also been explored in the context of computer vision, for instance, generalization to different visual distributions of the same concepts (domain adaptation) [Csurka 2017], or generalization across tasks [Zamir et al. 2018]. Generalization across concepts is a crucial part of zero-shot [Socher et al. 2013] and few-shot [Vinyals et al. 2016] learning. We study this particular dimension, concept generalization, whose goal is to transfer knowledge acquired on a set of *seen* concepts, to newly encountered *unseen* concepts as effectively as possible. Different from existing work, we take a systematic approach by considering the semantic similarity between seen and unseen concepts when measuring concept generalization.

Towards a structure of the concept space. One of the first requirements for rigorously evaluating concept generalization is structuring the concept space, in order to analyze the impact of concepts present during pretraining and transfer stages. However, previous work rarely discusses the particular choices of splits (seen vs. unseen) of their data, and random sampling of concepts remains the most common approach [Hariharan and Girshick 2017, Jayaraman and Grauman 2014, Lampert et al. 2009, Xian et al. 2018a]. A handful of methods leverage relations designed by experts. The WordNet graph [Miller 1995] for instance helps build dataset splits in Frome et al. [2013], Yosinski et al. [2014] and a domain-specific ontology is used to test cross-domain generalization [Guo et al. 2020, Wallace and Hariharan 2020]. These splits are however based on heuristics, instead of principled mechanisms built on semantic relationship between concepts as we do in this work.

Transfer learning evaluations. When it comes to evaluating the quality of visual representations, the gold standard is to benchmark models by solving diverse tasks such as classification, detection, segmentation and retrieval on many datasets [Caron et al. 2019, Chen et al. 2020a, Ericsson et al. 2021, Goyal et al. 2019, He et al. 2020, Kornblith et al. 2019, Zhai et al. 2019b]. The most commonly used datasets are IN-1K [Russakovsky et al. 2015],

Places [Zhou et al. 2017], SUN [Xiao et al. 2010], Pascal-VOC [Everingham et al. 2009], MS-COCO [Lin et al. 2014]. Such choices, however, are often made independently from the dataset used to train the visual representations, ignoring their semantic relationship.

In summary, semantic relations between pretraining and transfer tasks have been overlooked in evaluating the quality of visual representations. To address this issue, we present a controlled evaluation protocol that factors in such relations.

3.3 The ImageNet-CoG benchmark

Transfer learning performance is highly sensitive to the semantic similarity between concepts in the pretraining and the target datasets [Deselaers and Ferrari 2011, Yosinski et al. 2014]. Studying this relationship requires carefully constructed evaluation protocols: i) controlling which concepts a model has been exposed to during training (seen concepts), and ii) the semantic distance between these seen concepts and those considered for the transfer task (unseen concepts). As discussed earlier, current evaluation protocols severely fall short on handling these aspects. To fill this gap, we propose *ImageNet Concept Generalization (CoG)*—a benchmark composed of multiple image sets, one for pretraining and several others for transfer, curated in a controlled manner in order to measure the transfer learning performance of visual representations to sets of unseen concepts with increasingly distant semantics from the ones seen during training.

While designing this benchmark, we considered several important points. First, in order to exclusively focus on concept generalization, we need a controlled setup tailored for this specific aspect of generalization. In other words, we need to make sure that the only change between the pretraining and the transfer datasets is the set of concepts. In particular, we need the input image distribution (natural images) and the annotation process (which may determine the statistics of images [Torralba and Efros 2011]) to remain constant.

Second, to determine the semantic similarity between two concepts, we need an auxiliary knowledge base that can provide a notion of semantic relatedness between visual concepts. It can be manually defined with expert knowledge, *e.g.*, WordNet [Miller 1995], or automatically constructed, for instance by a language model, *e.g.*, word2vec [Mikolov et al. 2013b].

Third, the choice of the pretraining and target datasets is crucial. We need these datasets to have diverse object-level images [Berg and Berg 2009] and to be as less biased as possible, *e.g.*, towards canonical views [Mezuman and Weiss 2012].

Conveniently, the **IN-21K dataset** fulfills all these requirements. We therefore choose it as the source of images and concepts for our benchmark. IN-21K contains 14,197,122 curated

images covering 21,841 concepts, all of which are further mapped into synsets from the WordNet ontology, which we use to measure semantic similarity.

In the rest of this section, we first define the disjoint sets of seen and unseen concepts, then present our methodology to build different levels for evaluating concept generalization, and describe the evaluation protocol.

3.3.1 Seen concepts

We make a natural choice and use the 1000 classes from the ubiquitous IN-1K dataset [Russakovsky et al. 2015] as the set of our *seen* concepts. IN-1K is a subset of the IN-21K [Deng et al. 2009]. It consists of 1.28M images and has been used as the standard benchmark for evaluating novel computer vision architectures [He et al. 2016, Simonyan and Zisserman 2015, Szegedy et al. 2016, Touvron et al. 2021], regularization techniques [Shen and Savvides 2020, Verma et al. 2019, Yun et al. 2019, Zhang et al. 2018] as well as self- and semi-supervised models [Caron et al. 2020, Chen et al. 2020b, Grill et al. 2020, He et al. 2020, Yalniz et al. 2019].

Choosing IN-1K as the seen classes further offers several advantages. Future contributions, following standard practice, could train their models on IN-1K, and then simply evaluate generalization on our benchmark with their pretrained models. It also enables us to benchmark visual representations learned on IN-1K out-of-the box, using publicly available models (as shown in Sec. 3.4).

3.3.2 Selecting eligible unseen concepts

Prior to creating the concept generalization levels, we determine a set of *eligible* unseen concepts from IN-21K implementing the following steps.

- We start with the whole set of IN-21K concepts (21,841) of the Fall 2011 release[2] and exclude the ones from IN-1K, as they are the seen concepts and we are interested in concepts that are not seen during training.
- We remove all the concepts that are ancestors of these 1000 seen concepts in the WordNet [Miller 1995] hierarchy[3]. For instance, the concept "cat" is discarded since its child concept "tiger cat" is in IN-1K.

[2]Note that the recently released Winter 2021 ImageNet version shares the same set of images for all the unseen concepts selected in our benchmark with the Fall 2011 one. We refer the reader to Appendix A for further discussion on both the recent Winter 2021 release as well as a newer, blurred version of IN-1K.

[3]In order to get superior-subordinate relationships between the concepts, we use WordNet-3.0 (the version ImageNet [Deng et al. 2009] is built on) implementation in the NLTK library [Bird et al. 2009].

n00005787	n00288384	n00466377	n00466524	n00466630	n00474568	n00475014	n00475273	n00475403	n00483313
n00483409	n00483508	n00483848	n01314388	n01314663	n01314781	n01317294	n01317813	n01317916	n01318381
n01318894	n01321770	n01322221	n01323355	n01323493	n01323599	n01324431	n01324610	n01515303	n01517966
n01526521	n01862399	n01887474	n01888181	n02075612	n02152881	n02153109	n02156871	n02157206	n02236355
n02377063	n02377291	n02472987	n02475078	n02475669	n02759257	n02767665	n02771004	n03198500	n03300216
n03349771	n03393017	n03443005	n03680512	n04164406	n04193377	n04224543	n04425804	n04516354	n04979002
n06255081	n06272612	n06274760	n07942152	n08182379	n08242223	n08578517	n09828216	n10300303	n13918274

Table 3.1: WordNet IDs of the 70 concepts considered problematic, therefore removed from the eligible list of unseen concepts.

- It was shown that some of the concepts under the "person" sub-tree in IN-21K can be offensive or visually inappropriate, which may lead to undesirable behavior in downstream applications [Yang et al. 2020]. We thus exclude the entire "person" sub-tree.
- We discard concepts that are not leaf nodes in the WordNet subgraph defined by all so-far-eligible concepts. Formally, for any c_1 and c_2 in the set of unseen concepts, we discard c_1 if c_1 is a parent of c_2.
- In order to create levels whose size is comparable to IN-1K, following the design choices made for IN-1K [Russakovsky et al. 2015], we removed concepts with fewer than 782 images (note that any concept in IN-1K contains at least 782 images and 50 of those are used within the test set).
- Finally, we manually inspected the remaining unseen concepts and found 70 potentially problematic concepts, which may be considered to be offensive, or too ambiguous to distinguish. Examples of such concepts include the very generic "People" (any group of human beings, men or women or children, collectively) or "Orphan" (a young animal without a mother) concepts. The list of such manually discarded concepts is given in Tab. 3.1.

These requirements reduce the set of eligible *unseen* IN-21K concepts to 5146 categories.

3.3.3 Concept generalization levels

Our next step is defining a sequence of unseen concept sets, each with decreasing semantic similarity to the seen concepts in IN-1K. We refer to each one of these as a *concept generalization level*. They allow us to measure concept generalization in a controlled setting, *i.e.*, to consider increasingly difficult transfer learning scenarios.

Recall that IN-21K is built on top of the word ontology WordNet, where distinct concepts or synsets are linked according to their semantic relationships drafted by linguists. This enables the use of existing semantic similarity measures [Budanitsky and Hirst 2006] that exploit the graph structure of WordNet to capture the semantic relatedness of pairs of concepts. Following prior work [Deselaers and Ferrari 2011, Rohrbach et al. 2010], we use Lin similarity [Lin

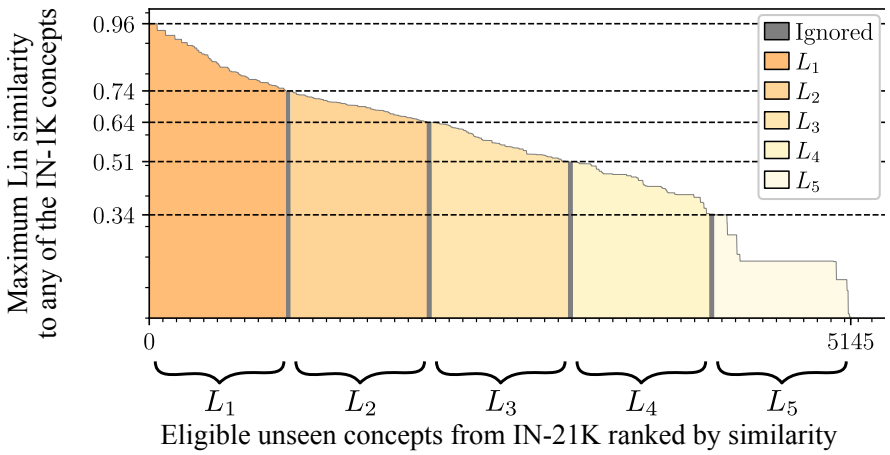


Figure 3.2: Concept generalization levels. We rank all the 5146 eligible IN-21K unseen concepts with respect to their similarity to IN-1K using Equation (3.2) and split the ranked list into 5 groups of 1000 concepts each. Each group defines a concept generalization level, each denoted by $L_{1/2/3/4/5}$. Gray-shaded areas correspond to concepts that are ignored.

1998] to define a concept-to-concept similarity. The Lin similarity between two concepts c_1 and c_2 is given by:

$$\text{sim}_{\text{Lin}}(c_1, c_2) = \frac{2 \times \text{IC}(\text{LCS}(c_1, c_2))}{\text{IC}(c_1) + \text{IC}(c_2)}, \tag{3.1}$$

where LCS denotes the lowest common subsumer of two concepts in the WordNet graph, and $\text{IC}(c) = -\log p(c)$ is the information content of a concept with probability $p(c)$ of encountering an instance of concept c in a specific corpus (in our case the subgraph of WordNet including all IN-21K concepts and their parents till the root node of WordNet: 'entity'). Following Resnik [1995], Rohrbach et al. [2011], we define $p(c)$ as the number of concepts that exist under c divided by the total number of concepts in the corpus. Probability of a concept ranges between $[0, 1]$ such that if c_2 is a parent of c_1 then $p(c_1) < p(c_2)$, and the probability of "Entity" becomes 1. An example of five concepts from IN-21K ranked by decreasing Lin similarity to the IN-1K concept "Tiger cat" is shown in Fig. 3.1(a).

We extend the above formulation to define the asymmetric similarity between the set of seen concepts from IN-1K, $\mathcal{C}_{\text{IN-1K}}$, and any unseen concept c as the maximum similarity between *any* concept from IN-1K and c:

$$\text{sim}_{\text{IN-1K}}(c) = \max_{\tilde{c} \in \mathcal{C}_{\text{IN-1K}}} (\text{sim}_{\text{Lin}}(c, \tilde{c})). \tag{3.2}$$

While designing our benchmark, we considered different semantic similarity measures before choosing Lin similarity. We explored other measures defined on the WordNet graph [Meng et al. 2013], such as the path-based Wu-Palmer [Wu and Palmer 1994] and the information

content-based Jiang-Conrath [Jiang and Conrath 1997]. We also considered semantic similarities based on Word2Vec representations [Mikolov et al. 2013b] of the titles and textual descriptions of the concepts. Our experiments with these alternative measures led to observations similar to the ones presented in Sec. 3.4 for Lin similarity. We refer the curious reader to the supplementary material for additional results with some of these measures.

With the similarity measure defined, our goal now is to group all eligible unseen concepts into multiple evaluation sets, which are increasingly challenging in terms of generalization. To ensure this, we would like the concepts contained in each consecutive set to be of decreasing semantic similarity to any concept from IN-1K. We achieve this by first ranking all unseen concepts with respect to their similarity to IN-1K using Equation (3.2). Then, we split the ranked list into groups of consecutive concepts as shown in Sec. 3.3.3; each group corresponds to a concept generalization level.

We design our levels to be comparable to IN-1K [Russakovsky et al. 2015], and therefore choose 1000 concepts per level. With 5146 eligible unseen concepts, we populate *five* sets. For increased diversity, we utilize the full span of the ranked list and end up with small gaps between levels (see supplementary material for more details). We denote the five concept generalization levels as $L_{1/2/3/4/5}$.

After selecting 1000 concepts for each level, we ensured that the image statistics are similar to those of IN-1K [Russakovsky et al. 2015], *i.e.*, we cap the number of images for each concept to a maximum of 1350 (1300 training + 50 testing). Note that we kept the same set of selected images per concept for all the experiments we performed in this work. We provide the complete list of image filenames in our code repository for reproducibility. In Fig. 3.3, we plot the number of images per concept for each of the five levels and for IN-1K. We note a minor class imbalance in all the generalization levels from these plots. To investigate if this imbalance had any effect on the observations of our benchmark, we further evaluated a subset of the models analyzed in Sec. 3.4 on a variant of the benchmark, where we randomly subsampled images from all the selected concepts to result in the same number of 732 training images, *i.e.*, on class-balanced levels. Apart from the overall reduced accuracy as a result of smaller datasets, this experiment produced similar results to the ones shown in Sec. 3.4, and all our observations continue to hold. We attribute this to the fact that imbalance is minimal.

3.3.4 Evaluation protocol

We now present the protocol for ImageNet-CoG, and summarize the metrics for the different experiments presented in Sec. 3.4. The benchmark consists of two phases. First, a feature extraction phase, where the model trained on IN-1K is used to extract features, followed

The ImageNet-CoG benchmark in a nutshell

Prerequisites:
A `model` pretrained on IN-1K
Sets of unseen concepts organized in levels $L_{1/2/3/4/5}$

Phase 1: Feature extraction
Use the `model` to extract image features for all image sets.

Phase 2: Evaluation
For the seen concepts (IN-1K) and for each level of unseen concepts ($L_{1/2/3/4/5}$), separately:

- Learn a linear classifier using all the training data
  < How resilient is my model to the semantic distance between seen and unseen concepts?>
- Learn a linear classifier using $N \in \{1,2,4,\ldots,128\}$ samples per concept.
  < How fast can my model adapt to new concepts?>

by the evaluation phase that is conducted on each level independently. An overview of the benchmark is presented in the gray box.

3.3.4.1 Phase 1: Feature extraction

We base our protocol on the assumption that *good* visual representations should generalize to new tasks with minimal effort, *i.e.*, without fine-tuning the backbones. Therefore, our benchmark only uses the pretrained backbones as feature extractors and decouples representation from evaluation. Concretely, we assume a model learned on the training set of IN-1K. We use this model as an encoder to extract features for images of IN-1K and of all the five levels $L_{1/2/3/4/5}$.

We extract features from the layer right before the classifiers from the respective models, following recent findings [Kolesnikov et al. 2019] that suggest that residual connections prevent backbones from overfitting to pretraining tasks. We ℓ_2-normalize the features and extract them offline: no data augmentation is applied when learning the subsequent classifiers.

3.3.4.2 Phase 2: Evaluation

We learn linear logistic regression classifiers for each level using all available training images. Since each level is by design a dataset approximately as big as IN-1K, we also learn linear classifiers on IN-1K with the same protocol; this allows us to compare performance across seen and unseen concepts. We also evaluate how efficiently models adapt when learning unseen concepts, *i.e.* how many samples they need to do so, by performing few-shot concept classification.

3.3.4.3 Metrics and implementation details

We report Top-1 accuracy for all the experiments. Absolute accuracy numbers are comparable across IN-1K and each level by construction, since all the levels share the same number of concepts and have training sets of approximately the same size. However, we mostly plot accuracy *relative to a baseline model*, for two reasons: (i) it makes the plots clearer and the differences easier to grasp, (ii) the performance range at each level is slightly different so it helps visualizing the trends better.

We perform SGD to train classifiers, with momentum=0.9 updates, using batches of size 1024, and apply weight decay regularization to parameters. To create the train/test split, we randomly select 50 samples as the test set for each concept and use the remaining ones (at least 732, at most 1300) as a training set. We use part of the training data for each level to optimize the hyper-parameters of the logistic regression (*i.e.*, learning rate and weight decay parameters).

We use Optuna [Akiba et al. 2019] to optimize the learning rate and weight decay hyper-parameters for every model and every level; we use 20% of the training sets as a validation set to find the best configuration and then re-train using the complete training set. We report results only on the test sets. We repeat the hyper-parameter selection 5 times with different seeds, and report the mean of the final scores. This means that, in each repetition, we take a different random subset of the training set as a validation set and start hyper-parameter tuning with different random pairs of hyper-parameters. Despite this stochasticity, the overall pipeline is quite robust, with standard deviation in most cases less than 0.2, therefore, not clearly visible in figures.

3.4 Evaluating models on ImageNet-CoG

We now present our large-scale experimental study which analyzes how different CNN-based and transformer-based visual representation models behave on our benchmark, following the evaluation protocol defined in the previous section. For clarity, we only highlight a subset of our experiments and provide additional results in the Appendix B.

3.4.1 Models

We choose 31 models to benchmark and present the complete list in Tab. 3.2. To ease comparisons and discussions, we split the models into the following four categories.

Architecture. We consider several architectures including CNN-based (a-VGG19 [Simonyan and Zisserman 2015], a-Inception-v3 [Szegedy et al. 2016], ResNet50, a-ResNet152 [He

Model	**Notes** (optionally # param. / amount of extra data)
Reference model: ResNet50	
ResNet50	Baseline model from the torchvision package (23.5M)
Architecture: Models with different backbone	
a-T2T-ViT-t-14	Visual transformer (21.1M)
a-DeiT-S	Visual transformer (21.7M)
a-DeiT-S-distilled	Distilled *a*-DeiT-S (21.7M)
a-Inception-v3	CNN with inception modules (25.1M)
a-NAT-M4	Neural architecture search model (7.6M)
a-EfficientNet-B1	Neural architecture search model (6.5M)
a-EfficientNet-B4	Neural architecture search model (17.5M)
a-DeiT-B-distilled	Bigger version of *a*-DeiT-S-distilled (86.1M)
a-ResNet152	Bigger version of ResNet50 (58.1M)
a-VGG19	Simple CNN architecture (139.6M)
Self-supervision: ResNet50 models trained in this framework	
s-SimCLR-v2	Online instance discrimination (ID)
s-MoCo-v2	ID with momentum encoder and memory bank
s-BYOL	Negative-free ID with momentum encoder
s-MoCHi	ID with negative pair mining
s-InfoMin	ID with careful positive pair selection
s-OBoW	Online bag-of-visual-words prediction
s-SwAV	Online clustering
s-DINO	Online clustering
s-BarlowTwins	Feature de-correlation using positive pairs
s-CompReSS	Distilled from SimCLR-v1 (with ResNet50x4)
Regularization: ResNet50 models with additional regularization	
r-MixUp	Label-associated data augmentation
r-Manifold-MixUp	Label-associated data augmentation
r-CutMix	Label-associated data augmentation
r-ReLabel	Trained on a "multi-label" version of IN-1K
r-Adv-Robust	Adversarially robust model
r-MEAL-v2	Distilled ResNet50
Use of web data: ResNet50 models using additional data	
d-MoPro	Trained on WebVision-V1 ($\sim 2\times$)
d-Semi-Sup	Pretrained on YFCC-100M ($\sim 100\times$), fine-tuned on IN-1K
d-Semi-Weakly-Sup	Pretrained on IG-1B ($\sim 1000\times$), fine-tuned on IN-1K
d-CLIP	Trained on WebImageText ($\sim 400\times$)

Table 3.2: List of models evaluated on ImageNet-CoG.

et al. 2016]), transformer-based (*a*-DeiT-S [Touvron et al. 2021], *a*-DeiT-S-distilled, *a*-DeiT-B-distilled, *a*-T2T-ViT-t-14 [Yuan et al. 2021a]) and neural architecture search (*a*-NAT-M4 [Lu et al. 2021], *a*-EfficientNet-B1 [Tan and Le 2019], *a*-EfficientNet-B4 [Tan and Le 2019]) backbones with varying complexities. We color-code the models in this category into two groups, depending on whether their number of parameters are comparable to ResNet50 (red) or not (orange); If they do, they are also directly comparable to all models

from the following categories.

Self-supervision. ResNet50-sized self-supervised models (in blue) include contrastive (*s*-SimCLR-v2 [Chen et al. 2020a,b], *s*-MoCo-v2 [Chen et al. 2020c, He et al. 2020], *s*-InfoMin [Tian et al. 2020b], *s*-MoCHi [Kalantidis et al. 2020], *s*-BYOL [Grill et al. 2020]), clustering-based (*s*-SwAV [Caron et al. 2020], *s*-OBoW [Gidaris et al. 2021], *s*-DINO [Caron et al. 2021]), feature de-correlation (*s*-BarlowTwins [Zbontar et al. 2021]), and distilled (*s*-CompReSS [Koohpayegani et al. 2020]) models.

Regularization. ResNet50-sized models with label regularization techniques (in purple) applied during the training phase include distillation (*r*-MEAL-v2 [Shen and Savvides 2020]), label augmentation (*r*-MixUp [Zhang et al. 2018], *r*-Manifold-MixUp [Verma et al. 2019], *r*-CutMix [Yun et al. 2019] and *r*-ReLabel [Yun et al. 2021]) and adversarial robustness (*r*-Adv-Robust [Salman et al. 2020]) models.

Use of web data. Models pretrained using additional web data with noisy labels are color-coded in green. This includes student-teacher models *d*-Semi-Sup [Yalniz et al. 2019] and *d*-Semi-Weakly-Sup [Yalniz et al. 2019], which are first pretrained on YFCC-100M [Thomee et al. 2016] (100x the size of IN-1K) and IG-1B [Mahajan et al. 2018] (1000x) and then fine-tuned on IN-1K. We also consider cross-modal *d*-CLIP [Radford et al. 2021a] pretrained on WebImageText (400x) with textual annotations, and noise tolerant tag prediction model *d*-MoPro pretrained on WebVision-V1 [Li et al. 2017] (2x). As it is not clear if YFCC-100M, IG-1B, WebImageText or WebVision-V1 contain images of the unseen concepts we selected in the levels, *models in this category are not directly comparable*.

We use publicly available models provided by the corresponding authors for all these approaches. All the models, with the exception of those in the use-of-web-data category, are only pretrained on IN-1K. We also use the best ResNet-50 backbones released by the authors for all the ResNet-based models. We use the vanilla ResNet50 (the version available in the torchvision package) as a reference point, which makes cross-category comparisons easier. We prefix models' names with the category identifiers for clarity.

When extracting features from these models, we first resize an image such that its shortest side becomes S pixels, then take a center crop of size $S \times S$ pixels. To comply with the testing schemes of the models, for all the backbones we set $S = 224$, except *a*-Inception-v3 [Szegedy et al. 2016] ($S = 299$), *a*-DeiT-B-distilled [Touvron et al. 2021] ($S = 384$), *a*-EfficientNet-B1 [Tan and Le 2019] ($S = 240$) and *a*-EfficientNet-B4 [Tan and Le 2019] ($S = 380$).

We also adapt their normalization schemes to be compatible with the data augmentation pipeline of the pretrained models. Concretely, we normalize each image by first dividing

Model	Feature Dim.
All models with ResNet50 [He et al. 2016] backbone	2048
a-T2T-ViT-t-14 [Yuan et al. 2021a]	384
a-DeiT-S [Touvron et al. 2021]	384
a-DeiT-B-distilled [Touvron et al. 2021]	768
a-NAT-M4 [Lu et al. 2021]	1536
a-EfficientNet-B1 [Tan and Le 2019]	1280
a-EfficientNet-B4 [Tan and Le 2019]	1792
a-VGG19 [Simonyan and Zisserman 2015]	4096

Table 3.3: Unique architectures used by the models we evaluate on ImageNet-CoG, and the dimensionality of the feature vectors we extract from these architectures.

them by 255 (so that each pixel value is in $[0,1]$), then applying mean and standard deviation normalization to the pixels, *i.e.*, subtracting $[0.485,0.456,0.406]$ from the RGB channels and diving them by $[0.229,0.224,0.225]$, respectively. Note that for *d*-CLIP [Radford et al. 2021a] we use mean $[0.481,0.457,0.408]$ and std $[0.268,0.261,0.275]$, and do not apply normalization for *s*-SimCLR-v2 [Chen et al. 2020b].

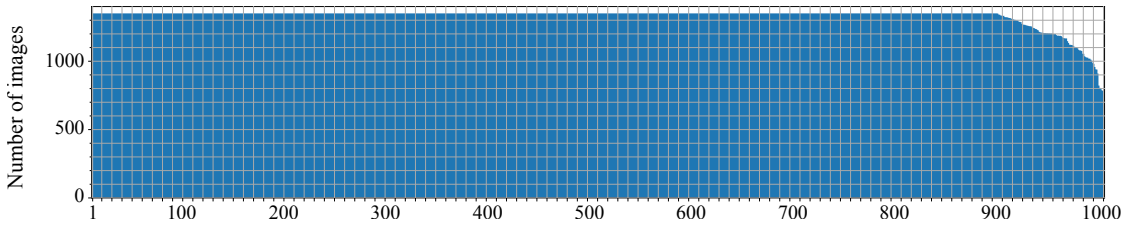
Tab. 3.3 lists the set of unique backbone architectures considered in our study, and the dimensionality of the produced feature representations. For all the architectures trained in a supervised way, we extract features from the penultimate layers, *i.e.*, before the last fully-connected layers making class predictions. For self-supervised learning methods, we follow the respective papers and extract features from the layer learned for transfer learning.

3.4.2 Results

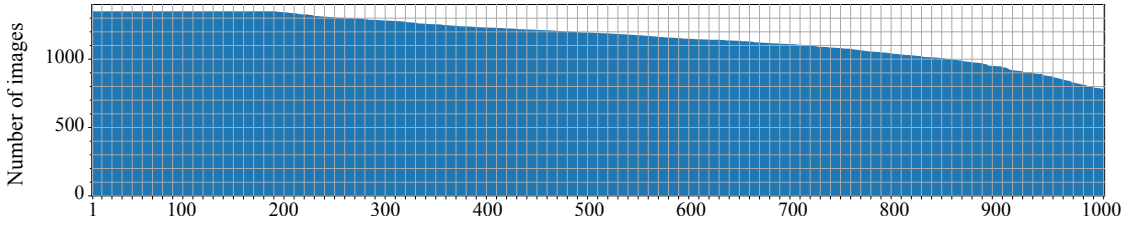
We measure image classification performance on IN-1K and each of the concept generalization levels $L_{1/2/3/4/5}$ of ImageNet-CoG for the 31 models presented above, using a varying number of images per concept. These experiments allow us to study (i) how classification performance changes as we semantically move away from the seen concepts (Sec. 3.4.2.1), and (ii) how fast models can adapt to unseen concepts (Sec. 3.4.2.2). We refer the reader to Sec. 3.3.4 for the justification of our protocol and the choice of metrics.

3.4.2.1 Generalization to unseen concepts

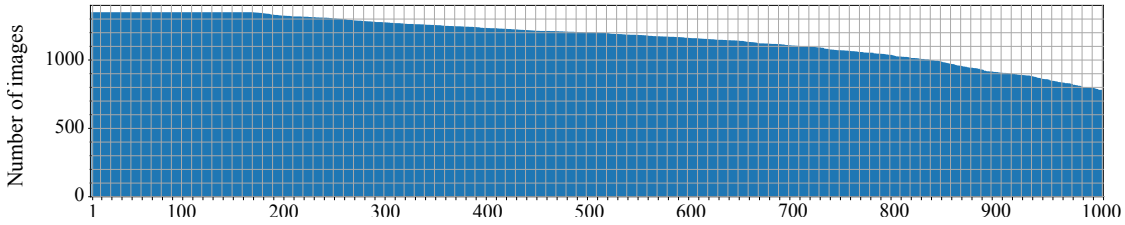
We report the performance of linear classifiers learnt with all the training data in Fig. 3.4. In Fig. 3.4(a) we report Top-1 accuracy for all models and levels, while Fig. 3.4(b)-(e) present performance relative to the baseline ResNet50 across the 4 model categories. Our main observations are as follows.



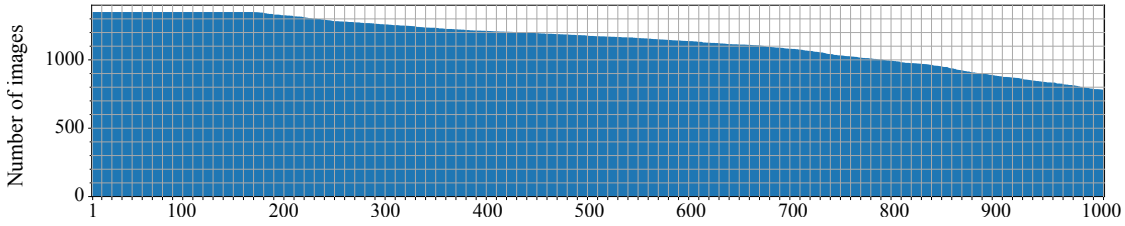
(a) IN-1K [Russakovsky et al. 2015]



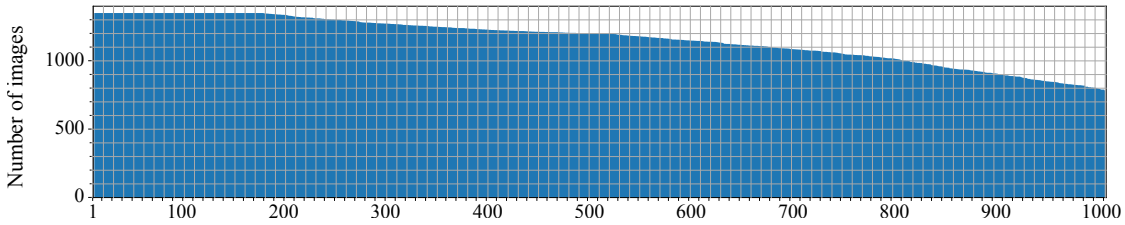
(b) Lin L_1



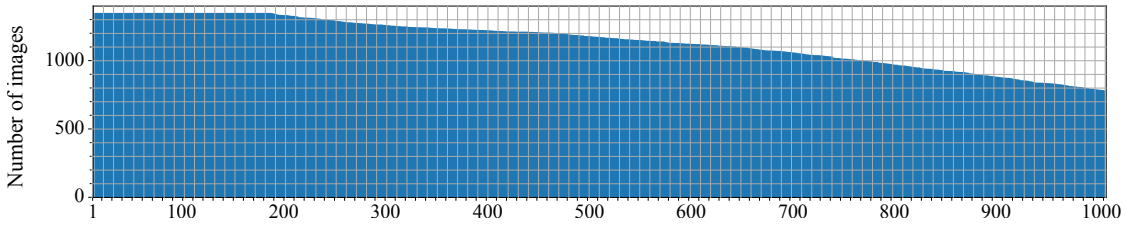
(c) Lin L_2



(d) Lin L_3

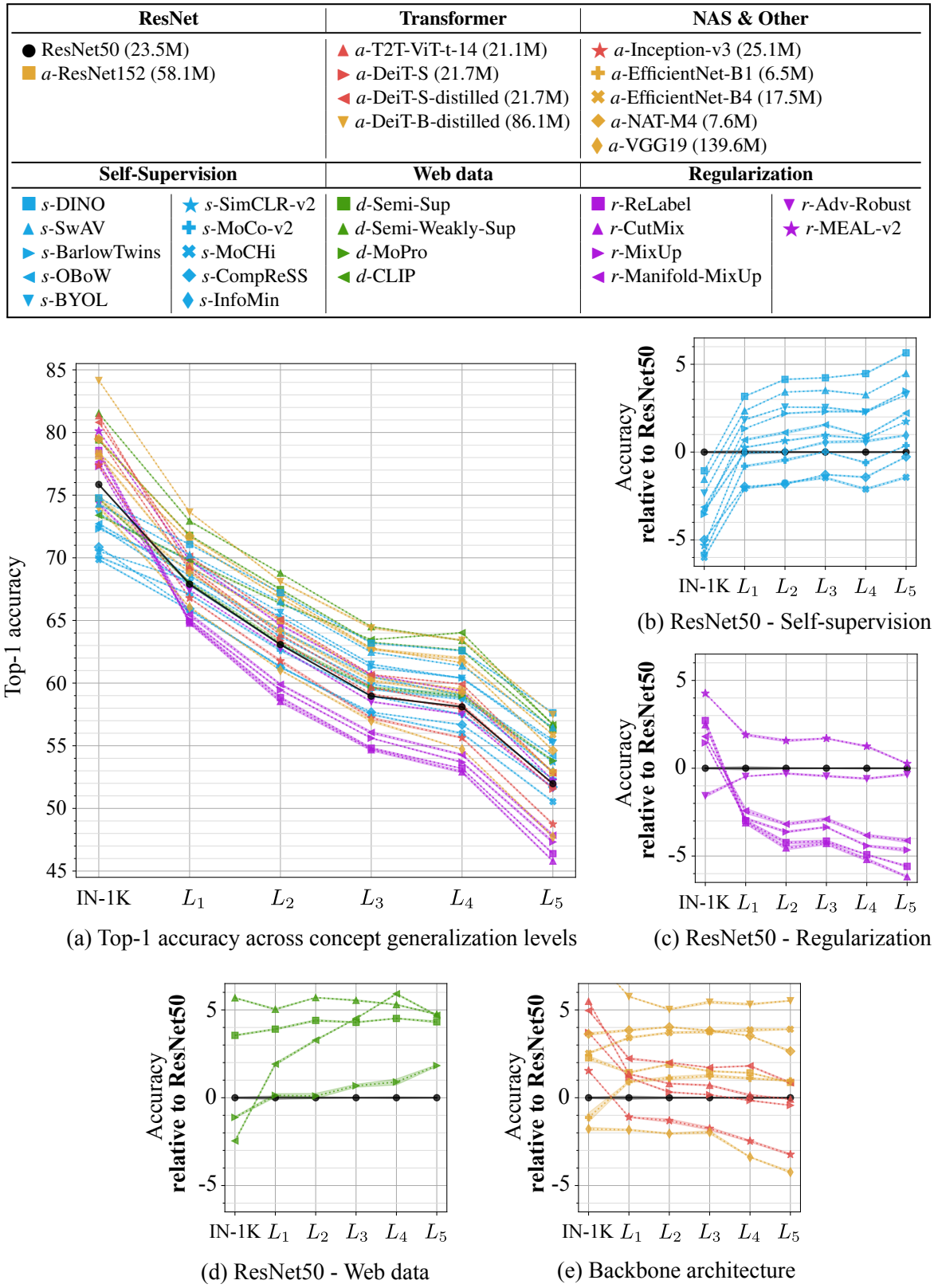


(e) Lin L_4



(f) Lin L_5

Figure 3.3: The number of images per concept for IN-1K [Russakovsky et al. 2015] and each of the concept generalization levels obtained by Lin similarity. We end up with 1.17M, 1.17M, 1.15M, 1.16M, 1.14M images in total for levels $L_{1/2/3/4/5}$ respectively. Note that IN-1K has 1.33M images in total.



(a) Top-1 accuracy across concept generalization levels

(b) ResNet50 - Self-supervision

(c) ResNet50 - Regularization

(d) ResNet50 - Web data

(e) Backbone architecture

Figure 3.4: Linear classification on ImageNet-CoG. Top-1 accuracies for all the 31 models listed in Tab. 3.2 after training logistic regression classifiers on IN-1K and each level $L_{1/2/3/4/5}$. (a) Absolute Top-1 accuracy on all levels. (b)-(e) accuracy relative to the baseline ResNet50 for all the models, split across the four model categories presented in Sec. 3.4.1.

* ***It is harder to generalize to semantically distant concepts.*** The absolute performance of all models monotonically decreases as we move away semantically from IN-1K. This implies that transfer learning becomes more and more challenging on levels from L_1 to L_5, *i.e.*, as we try to distinguish concepts that are further from the training ones.

* ***Self-supervised models excel at concept generalization.*** Many recent self-supervised models (*s*-DINO, *s*-SwAV, *s*-BYOL, *s*-OBoW and *s*-SimCLR-v2) outperform ResNet50 on all levels. In general, we see that the performance gaps between ResNet50 and self-supervised models progressively shift in favor of the latter (Fig. 3.4(b)). Surprisingly, from Fig. 3.4(a) we also see that a ResNet50 trained with *s*-DINO competes with the top-performing models on L_5 *across all categories and model sizes*. This shows that augmentation invariances learned by the model transfer well to images of unseen concepts.

* ***Visual transformers overfit more to seen concepts*** (for models with as many parameters as ResNet50). The top-performing model of the study overall is *a*-DeiT-B-distilled, a large visual transformer. However, for the same number of parameters as ResNet50, we see that the large gains that visual transformers like *a*-DeiT-S and *a*-T2T-ViT-t-14 exhibit on IN-1K are practically lost for unseen concepts (red lines in Fig. 3.4(e)). In fact, both end up performing slightly worse than ResNet50 on L_5.

* ***Using noisy web data highly improves concept generalization.*** Weakly-supervised models *d*-Semi-Sup, *d*-Semi-Weakly-Sup and *d*-CLIP pretrained with roughly 100x, 1000x, and 400x more data than IN-1K exhibit improved performance over ResNet50 on all levels (Fig. 3.4(d)). It is worth re-stating, however, that since their datasets are web-based and much larger than IN-1K, we cannot confidently claim that concepts in our levels are indeed *unseen* during training. Results on this model category should therefore be taken with a pinch of salt.

* ***Model distillation generally improves concept generalization performance.*** We see that distilled supervised models *r*-MEAL-v2 and *a*-DeiT-S-distilled consistently improve over their undistilled counterparts on all levels (Fig. 3.4(c) and (e)). However, these gains decrease progressively, and for L_5 performance gains over the baseline are small. It is also worth noting that adversarial training (*r*-Adv-Robust) does not seem to hurt concept generalization.

* ***Neural architecture search (NAS) models seem promising for concept generalization.*** All NAS models we evaluate (*a*-EfficientNet-B1, *a*-EfficientNet-B4 and *a*-NAT-M4) exhibit stable gains over the baseline ResNet50 on all levels (Fig. 3.4(e)), showing good concept generalization capabilities. Among them, *a*-NAT-M4, a NAS model tailored for transfer learning with only 7.6M parameters achieves particularly impressive performance over all levels including IN-1K.

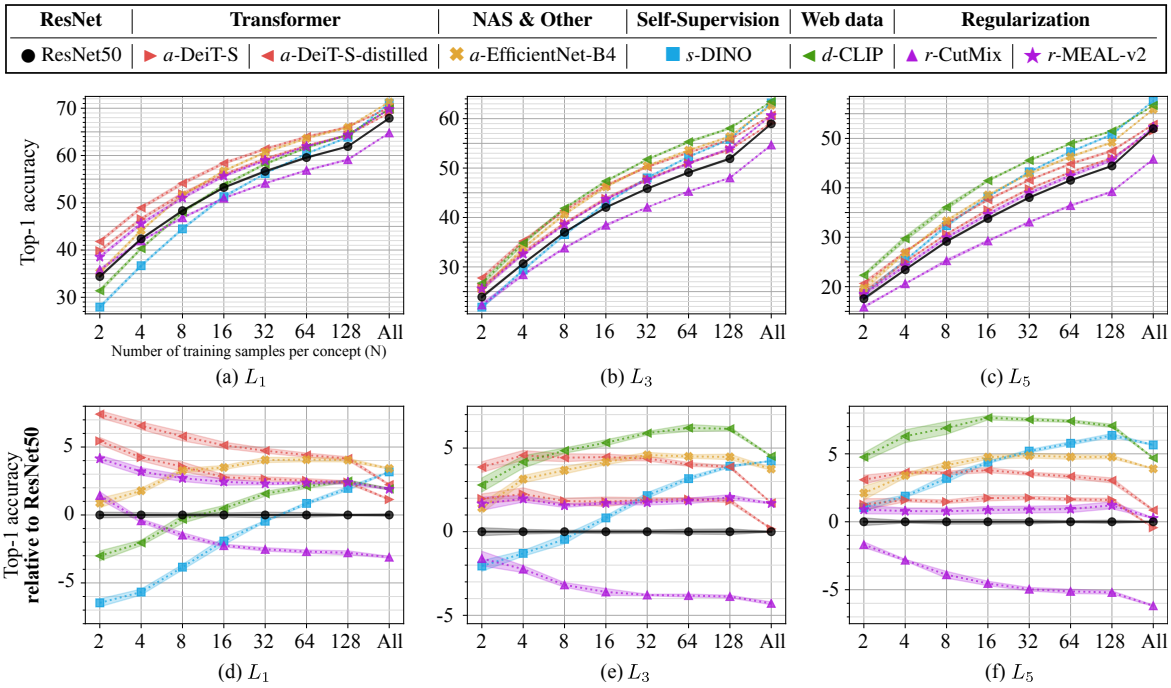


Figure 3.5: Few-shot linear classification on ImageNet-CoG. Top-1 accuracies for a subset of the models listed in Tab. 3.2 after training logistic regression classifiers on L_1, L_3, L_5 using $N = \{2, 4, 8, 16, 32, 64, 128\}$ training samples per concept. Performance when using all the samples is also shown for reference. (a)-(c): Absolute Top-1 accuracy. (d)-(f) accuracy relative to the baseline ResNet50. *The complete set of results for all the 31 models and levels is in the supplementary material.*

* ***Label-associated augmentation techniques deteriorate concept generalization performance.*** Although methods like *r*-MixUp, *r*-Manifold-MixUp, *r*-ReLabel and *r*-CutMix exhibit strong performance gains over ResNet50 on IN-1K, *i.e.*, for concepts seen during training, Fig. 3.4(c) shows that such gains do not transfer when generalizing to unseen ones. They appear to overfit more to the seen concepts.

* ***What are the top-performing models overall for concept generalization?*** From Fig. 3.4(a) we see that better and larger architectures and models using additional data are on top for L_3-L_5. However, it is impressive how *s*-DINO, a contrastive self-supervised model, is among the top methods, outperforming the vast majority of models at the most challenging levels.

3.4.2.2 How fast can models adapt to unseen concepts?

We now study few-shot classification, *i.e.*, training linear classifiers with $N = \{2, 4, 8, 16, 32, 64, 128\}$ samples per concept. For clarity, we selected a subset of the models and in Fig. 3.5 we present

their performance on L_1, L_3 and L_5. The complete set of results for all models and levels is given in Appendix B. We discuss the most interesting observations from Fig. 3.5 below.

* ***Transformer-based models are strong few-shot learners.*** Transformer-based models exhibit consistent gains over ResNet50 on all levels when $N \leq 128$. Despite the fact that performance gains from transformers diminish when using all available images on L_5, they exhibit a consistent 3-4% accuracy gain over ResNet50 for $N \leq 128$ (Fig. 3.5(f)).

* ***Model Distillation and Neural Architecture Search (NAS) exhibit consistent gains also in low-data regimes.*** The NAS-based *a*-EfficientNet-B4 model exhibits consistently higher performance than ResNet50 on all levels for all N. The same stands for the distilled *r*-MEAL-v2 and *a*-DeiT-S-distilled that are also consistently better than their undistilled counterparts for all N and all levels.

* ***Bigger models and additional web data help at few-shot learning.*** This is an observation from the extended set of figures (see Appendix B). Bigger models have consistent gains in low-data regimes. The same stands for models with additional web data. Moreover, as we go towards semantically dissimilar concepts, *a*-NAT-M4 outperforms all other methods and it even challenges the much bigger *a*-DeiT-B-distilled model.

3.5 Further analysis on ImageNet-CoG

In this section, we investigate some of the properties and design choices of ImageNet-CoG. More specifically, in Sec. 3.5.1 we compare linear evaluation to fine-tuning models on the ImageNet-CoG levels. Then, in Sec. 3.5.2 we re-group the 5000 unseen concepts in ImageNet-CoG by using another semantic similarity measure based on pure textual information about concepts, *i.e.*, word2vec similarity. Finally, in Sec. 3.5.3, we briefly discuss the potential noise from missing labels in ImageNet-CoG.

3.5.1 What if we fine-tuned the backbones?

Our benchmark and evaluation protocol are based on the assumption that good visual representations should generalize to different tasks with minimal effort. In fact, we explicitly choose to decouple representation learning from training classifiers and consider frozen/pretrained backbones as feature extractors. We then evaluate how well pretrained representations transfer to concepts *unseen* during representation learning. Fine-tuning the models would therefore go against the main premise of our benchmark: After fine-tuning all concepts are "seen" during representation learning, *i.e.*, the feature spaces can now be adapted. It would then be unclear: Are we measuring the generalization capabilities of the pretraining strategy or of the fine-tuning process? How much does the latter affect generalization? We consider

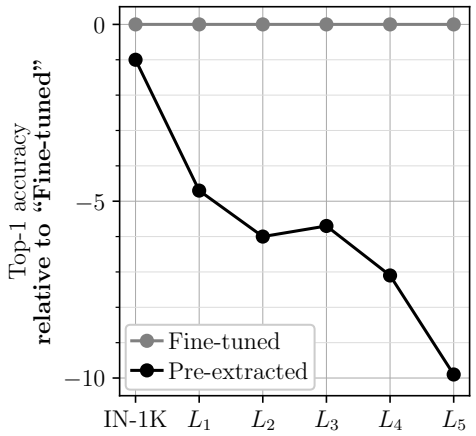


Figure 3.6: Comparison of training linear classifiers on **pre-extracted features** *vs.* **fine-tuning** backbones on each level. Y-axis shows the Top-1 accuracies obtained **relative** to the accuracy of the fine-tuned models.

such questions out of the scope of our study. In fact, learning linear classifiers on top of pre-extracted features additionally allows us to exhaustively optimize hyper-parameters for all the methods and levels, making sure that comparisons are fair across all models.

Measuring performance *relative to fine-tuning*, would however verify that the observed performance drops are due to increasing semantic distance and not variabilities across the levels. To this end, we fine-tune ResNet50 (pretrained on IN-1K) on IN-1K and on levels $L_{1/2/3/4/5}$ separately. Then we compare their performance with the protocol we choose for our benchmark, *i.e.* the case where we learn linear classifiers on top of pre-extracted features. In Fig. 3.6, we show the *relative* scores of the linear classifiers on top of pre-extracted (labeled as "Pre-extracted") against fine-tuned ResNet50s (labeled as "Fine-tuned").

We observe that pre-extracted features become less and less informative for unseen concepts as we move from IN-1K to L_5, supporting our main assumption that semantically less similar concepts are harder to classify.

3.5.2 word2vec as an alternative semantic similarity

One of the requirements for studying concept generalization in a controlled manner is a knowledge base that provides the semantic relatedness of any two concepts. As IN-21K is built on the concept ontology of WordNet [Miller 1995], in Sec. 3.3.3 we leverage its graph structure, and propose a benchmark where semantic relationships are computed with the Lin measure [Lin 1998].

As mentioned in Sec. 3.3, the WordNet ontology is hand-crafted, requiring expert knowledge. Therefore similarity measures that exploit this ontology (such as Lin) are arguably

reliable in capturing the semantic similarity of concepts. However, it could also be desirable to learn semantic similarities automatically, for instance, using other knowledge bases available online such as Wikipedia. In this section, we look at if such knowledge bases could be used in building our ImageNet-CoG.

With this motivation, we turn our attention to semantic similarity measures that can be learned over textual data describing the IN-21K concepts. Note that each IN-21K concept is provided with a name[4] and a short description[5]. The idea is to use this information to determine the semantic relatedness of any two concepts.

To this end, we leverage language models to map the textual description of any concept into an embedding vector, such that the semantic similarity between two concepts can be measured as the similarity between their representations in that embedding space. We achieve this through the skip-gram language model [Mikolov et al. 2013b], which has been extensively used in many natural language processing tasks, to extract "word2vec" representations of all concepts. However, we note that the name of many IN-21K concepts are *named entities* composed of multiple words, yet the vanilla skip-gram model tokenizes a textual sequence into words. We address this issue following Yamada et al. [2016] that learns a skip-gram model by taking into account such named entities. Specifically, we use the skip-gram model trained on Wikipedia[6] by the Wikipedia2Vec software [Yamada et al. 2020].

We compute the word2vec embeddings of IN-21K concepts as follows. Firstly, we combine the names and descriptions of all concepts and learn tf-idf weights for each unique word. Secondly, for each concept, we compute two word2vec representations: one for the concept name, and one for the concept description, by averaging the word2vec representations of the words that compose them. These two average vectors are added and used as the final word2vec representation of the concept. Finally, as the semantic similarity measure, we simply use the cosine similarity between the word2vec representations of two concepts:

$$\text{sim}_{\text{w2v}}(c_1, c_2) = \frac{\omega_{c_1} \top \omega_{c_2}}{\|\omega_{c_1}\| \cdot \|\omega_{c_2}\|}, \tag{3.3}$$

where ω_c denotes the word2vec representation of concept c.

Recall that in Sec. 3.3.3, first we rank the 5146 eligible unseen concepts in IN-21K (which remain after our filtering), w.r.t. their Lin similarity to the concepts in IN-1K. Then, we subsample 5000 concepts to construct concept generalization levels. To create another benchmark based on the textual information of the concepts as described above, we could repeat

[4] http://www.image-net.org/archive/words.txt
[5] http://www.image-net.org/archive/gloss.txt
[6] April 2018 version of the English Wikipedia dump.

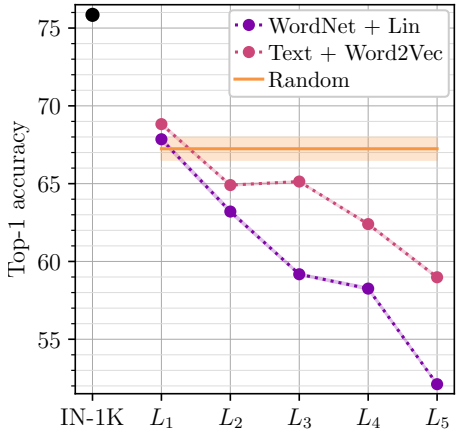


Figure 3.7: Semantic similarities of the concepts captured by (i) Lin similarity [Lin 1998] on WordNet graph [Miller 1995] and (ii) cosine similarity of word2vec embeddings [Yamada et al. 2020] extracted from textual descriptions of concepts, *vs.* **visual similarities** encoded by ResNet50, on IN-1K and generalization levels $L_{1/2/3/4/5}$ of ImageNet-CoG. We report the performance of linear logistic regression classifiers trained on features extracted from the global average pooling layer of ResNet50. The orange line shows results obtained on 1000 *random* unseen concepts (line represents the mean accuracy obtained over 15 random splits).

this procedure by replacing Lin similarity with the cosine similarity we defined in Equation (3.3). However, this could select a different sub-set of 5000 concepts, which, in turn, would produce two benchmarks with different sets of unseen concepts. To prevent this, we re-rank the 5000 concepts selected by the Lin similarity, based on their text-based cosine similarity to IN-1K concepts. Then we simply divide the re-ordered concepts into 5 disjoint sequential sets.[7]

We compare the two benchmarks constructed with different knowledge bases (*i.e.*, using the WordNet graph *vs.* textual descriptions) for our baseline model ResNet50 [He et al. 2016] that is pretrained on the seen concepts (IN-1K) for image classification, following our standard protocol. Concretely, first, we extract image features from the penultimate layer of the ResNet50, then we train linear classifiers on each concept domain separately.

We report results in Fig. 3.7 for the two benchmarks as well as randomly selected sub-sets of 1000 concept each. We see that the benchmark constructed using the WordNet ontology [Miller 1995] and the Lin similarity [Lin 1998] yield much more challenging concept generalization levels than the one obtained using textual data and a skip-gram language model [Yamada et al. 2020] pretrained on Wikipedia. This is especially visible when comparing classification performance on the levels $L_{3/4/5}$ produced by each technique. We argue

[7]Note that, given that the percentage of discarded concepts is very small (less than 3%, as 146 concepts are discarded from the 5146 eligible ones), this choice has minimal impact anyway.

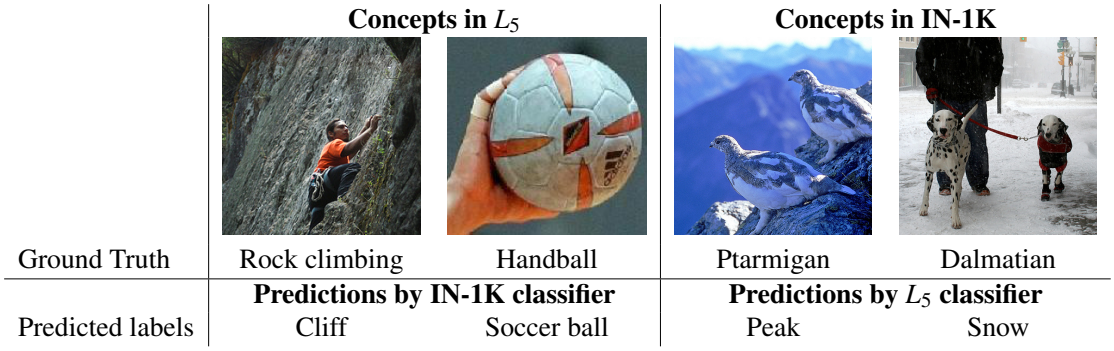


Figure 3.8: Illustration of the label noise in ImageNet-CoG.

that this could be due to the fact that WordNet is an ontology hand-crafted by experts and is able to better approximate the semantic similarity of two concepts compared to the learned skip-gram model. We see that, for a given level L_i, WordNet combined with Lin similarity manages to gather concepts that are harder to discriminate and that the resulting classification performance is lower. This experiment, however, shows that it is possible to create a similar benchmark using automatically produced semantic similarity scores, the main alternative in the absence of any reliable hand-crafted ontology.

3.5.3 Potential label noise in ImageNet-CoG

It has been shown recently [Yun et al. 2021] that IN-1K has missing-label noise. We can assume this extends to ImageNet-21K (IN-21K). Unfortunately, this type of noise is really difficult to correct and beyond the scope of our benchmark. However, we devise an experiment to get a sense of how much this noise could be.

Concretely, we take ResNet-50 classifiers trained for L_5 and apply them to all the images of the IN-1K val set and vice versa (*i.e.*, apply IN-1K classifiers on L_5 val). After inspecting samples that are predicted with very high confidence (> 0.99, about 2.7% of the images), we observe *several* cases where an unseen concept has (arguably) been seen during training without its label. Some examples are shown in Fig. 3.8. Given the low percentage of very confident matches and the fact that [Yun et al. 2021] does not show a big change in performance after re-training with the noise corrected, we believe that this type of labeling noise does not significantly affect our findings.

3.6 Conclusion

In this work, we studied concept generalization through the lens of our new ImageNet-CoG benchmark. It is designed to be used out-of-the-box with IN-1K pretrained models. We

evaluated a diverse set of 31 methods representative of the recent advances in visual representation learning.

Our extensive analyses show that self-supervised learning produces representations that generalize surprisingly better than any supervised model with the same number of parameters. We see that the current transformer-based models appear to overfit to seen concepts, unlike neural architecture-search-based models. The latter outperform several other supervised learning models with far less parameters.

We also studied how fast models can adapt to unseen concepts by learning classifiers with only a few images per class. In this setting, we verify that visual transformers are strong few-shot learners, and show how distillation and neural architecture search methods achieve consistent gains even in low-data regimes.

We envision ImageNet-CoG to be an easy-to-use evaluation suite to study one of the most important aspects of generalization in a controlled and principled way.

Chapter 4

Improving the generalization of supervised learning models

Contents

4.1 Introduction 58

4.2 Related work 60

4.3 An improved training setup for supervised learning 62

4.4 Experiments 64

4.4.1 Analysis of component design and hyper-parameters 67

4.4.2 Analysis of learned features, class weights and prototypes 70

4.4.3 Pushing the envelope of training-versus-transfer performance 72

4.4.4 Evaluations on the additional transfer datasets 74

4.5 Conclusion 75

Having discussed the evaluation aspect of visual representation learning in the previous chapter, we now turn our attention to the modeling aspect, and focus on developing models to learn representations "useful" for transfer tasks. Building on some of the observations made in the previous chapter, we propose an improved setup for training supervised models that learn transferable representations on ImageNet-1K. The work presented in this chapter[1] was accepted at the International Conference on Learning Representations (ICLR) in 2023 [Sariyildiz et al. 2023b].

More specifically, we consider the problem of training a deep neural network on a given classification task, *e.g.*, ImageNet-1K (IN-1K), so that it excels at both the training task as well as at other (future) transfer tasks. These two seemingly contradictory properties impose a trade-off between improving the model's generalization and maintaining its performance on the original task. Models trained with self-supervised learning tend to generalize better than their supervised counterparts for transfer learning; yet, they still lag behind supervised models on IN-1K. In this work, we propose a supervised learning setup that leverages the best of both worlds. We extensively analyze supervised training using multi-scale crops for data

[1] https://europe.naverlabs.com/trex

augmentation and an expendable projector head, and reveal that the design of the projector allows us to control the trade-off between performance on the training task and transferability. We further replace the last layer of class weights with class *prototypes* computed on the fly using a memory bank and derive two models: **t-ReX** that achieves a new state of the art for transfer learning and outperforms top methods such as DINO and PAWS on IN-1K, and **t-ReX*** that matches the highly optimized RSB-A1 model on IN-1K while performing better on transfer tasks.

4.1 Introduction

Deep convolutional neural networks trained on large annotated image sets like ImageNet-1K (IN-1K) [Russakovsky et al. 2015] have shown strong generalization properties. This motivated their application to a broad range of transfer tasks including the recognition of concepts that are not encountered during training [Donahue et al. 2014, Razavian et al. 2014].

Recently, models trained in a self-supervised learning (SSL) framework have become popular due to their ability to learn without manual annotations, as well as their capacity to surpass supervised models in the context of transferable visual representations. SSL models like MoCo [He et al. 2020], SwAV [Caron et al. 2020], BYOL [Grill et al. 2020] or DINO [Caron et al. 2021] exhibit stronger transfer learning performance than models [Wightman et al. 2021] trained on the same data with annotations [Sariyildiz et al. 2021].

This achievement is on the one hand exciting, as SSL approaches do not require an expensive and error-prone annotation process, but also seemingly counter-intuitive [Wang et al. 2022b] as it suggests that access to additional information, *i.e.*, image labels, actually hinders the generalization properties of a model. Models learned via SSL are however not able to match their supervised counterparts on IN-1K classification, *i.e.*, on the concepts seen during training. Top-performing SSL and semi-supervised methods like DINO [Caron et al. 2021] or PAWS [Assran et al. 2021] still result in 3-5% lower Top-1 accuracy compared to optimized supervised models such as RSB-A1 [Wightman et al. 2021].

In this work, we argue that access to more information (in the form of manual annotations) should not hurt generalization, and we seek to improve the transferability of encoders learned in a supervised manner, while retaining their state-of-the-art performance on the supervised training task. The mismatch observed between IN-1K and transfer performance suggests that this goal is not trivial. It has been shown, for example, that popular regularization techniques such as Label Smoothing [Szegedy et al. 2016], Dropout [Srivastava et al. 2014] or CutMix [Yun et al. 2019], which improve IN-1K performance, actually lead to less transferable representations [Kornblith et al. 2021, Sariyildiz et al. 2021], and that representations learned on top of models underfitting their original task transfer better [Zhang et al. 2022].

We identify two key training components from the most successful SSL approaches that may lead to more transferable representations: multi-crop data augmentation [Caron et al. 2020] and the use of an expendable projector head, *i.e.*, an auxiliary module added after the encoder during training and discarded at test time [Chen et al. 2020a]. We study the impact of these two components on the transfer performance together with the performance on the training task, and present novel insights on the role of the projector design in this context. Furthermore, inspired by recent work on supervised learning [Feng et al. 2022, Khosla et al. 2020], we introduce *Online Class Means*, a memory-efficient variant of the Nearest Class Means classifier [Mensink et al. 2012] that computes class prototypes in an "online" manner with the help of a memory queue. This further increases performance. We perform an extensive analysis on how each component affects the learned representations, and look at feature sparsity and redundancy as well as intra-class distance. We also study the training dynamics and show that class prototypes and classifier weights change in different ways across iterations.

We single out the two ResNet50 instantiations that perform best at one of the two dimensions (transfer learning and IN-1K), denoted as **t-ReX** and **t-ReX***. **t-ReX** exceeds the state-of-the-art transfer learning performance of DINO [Caron et al. 2021] or PAWS [Assran et al. 2021] and still performs much better than these two on IN-1K classification. **t-ReX*** outperforms the state-of-the-art results of RSB-A1 [Wightman et al. 2021] on IN-1K while generalizing better to transfer tasks. We visualize the performance of these two selected models, together with those of other top-performing configurations from our setup in Sec. 4.4, and compare it to state-of-the-art supervised, semi-supervised and self-supervised learning methods, across two dimensions: IN-1K accuracy and mean transfer accuracy across 13 transfer tasks. This intuitively conveys how the proposed training setup *pushes the envelope* of the training-versus-transfer performance trade-off (from the "Previous SotA" region, to the "New SotA" one in Fig. 4.7) and offers strong pretrained visual encoders that future approaches could build on.

Contributions. We propose a supervised training setup that incorporates multi-crop data augmentation and an expendable projector and can produce models with favorable performance both on the training task of IN-1K and on diverse transfer tasks. We thoroughly ablate this setup and reveal that the design of the projector allows to control the performance trade-off between these two dimensions, while a number of analyses of the features and class weights give insights on how each component of our setup affects the training and learned representations. We also introduce *Online Class Means*, a prototype-based training objective that increases performance even further and gives state-of-the-art models for transfer learning (**t-ReX**) and IN-1K (**t-ReX***).

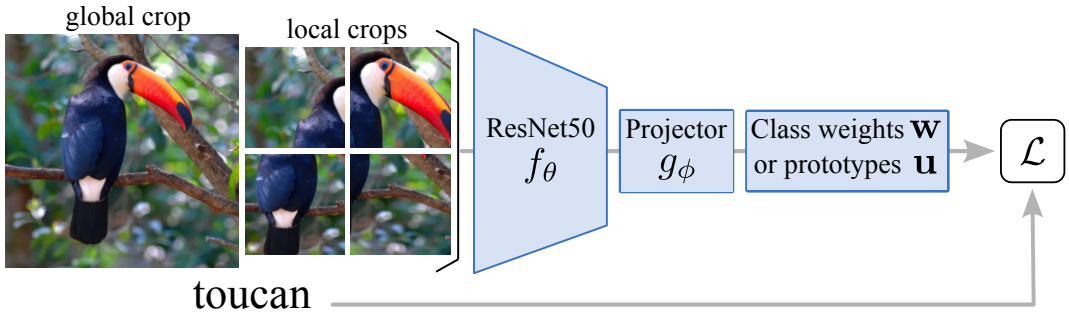
4.2 Related work

Visual representations learned by deep networks for IN-1K classification can transfer to other tasks and datasets [Donahue et al. 2014, Razavian et al. 2014]. This generalization capability of networks has motivated researchers to propose practical approaches for measuring transfer learning [Goyal et al. 2019, Pándy et al. 2022, Zhai et al. 2019a] or contribute to a formal understanding of generalization properties [Kornblith et al. 2019, Tripuraneni et al. 2020, Yosinski et al. 2014]. Recent work in this context [Kornblith et al. 2021, Sariyildiz et al. 2021] shows that the best representations for IN-1K are not necessarily the ones transferring best. For instance, some regularization techniques or loss functions improving IN-1K classification lead to underwhelming transfer results. A parallel line of work based on self-supervised learning [Caron et al. 2020, Chen et al. 2020a, Grill et al. 2020] focuses on training models without manual labels, and demonstrates their strong generalization capabilities to many transfer datasets, clearly surpassing their supervised counterparts [Sariyildiz et al. 2021]. Yet, as expected, SSL models are no match to the supervised models on the IN-1K classification task itself.

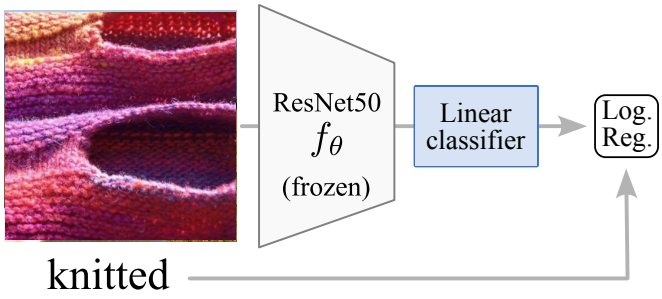
A few approaches tackle the task of training supervised models that also transfer well and share motivation with our work. SupCon [Khosla et al. 2020] extends SimCLR [Chen et al. 2020a] using image labels to build positive pairs. As such, its formulation is close to neighborhood component analysis (NCA) [Goldberger et al. 2004]. It circumvents the need for large batches by adding a momentum and a memory similar to MoCo [He et al. 2020]. Supervised-MoCo [Zhao et al. 2021b] filters out false negatives in the memory bank of MoCo using image labels, while LOOK [Feng et al. 2022] modifies the NCA objective to only consider the closest neighbors of each query image. We experimentally observe that our model design leads to better transfer than all these works.

In this work, we propose an effective training setup, which leverages multi-crop augmentation [Caron et al. 2020] and an expendable projector head [Chen et al. 2020a], two key components in many successful SSL approaches. Creating multiple augmented versions (a.k.a. crops) of images in a batch was first proposed by Hoffer et al. [2020]. Caron et al. [2020] further consider crops with different scales and resolutions in a self-supervised learning setting, creating challenging views of an image for which the model is encouraged to learn consistent representations [Assran et al. 2021, Caron et al. 2021]. Recent work argues that multi-crop increases representation variance, is useful for online self-distillation [Wang et al. 2022a], and improves vision and language pretraining [Ko and Gu 2022]. We show that multi-crop over different resolutions works out-of-the-box also for supervised training on IN-1K.

Using features from intermediate layers of networks has been considered before, *e.g.*, for



(a) Supervised learning using multi-crop and a projector.



(b) Transfer learning with a frozen model.

Figure 4.1: Our proposed supervised learning setup borrows multi-crop [Caron et al. 2020] and projectors [Chen et al. 2020a] from SSL to train on IN-1K (*top*). The projector g is discarded after training, and the ResNet backbone f is used as a feature extractor in combination with a linear classifier trained for each task, *e.g.* for texture classification on DTD [Cimpoi et al. 2014] (*bottom*).

training object detectors [Lin et al. 2017] and image classification models [Lee et al. 2015], or evaluating the transferability of individual layers [Gidaris et al. 2018, Zhang et al. 2016] or groups of layers [Evci et al. 2022]. However, selecting optimal layers for each problem is infeasible due to the computational nature of this selection. SimCLR [Chen et al. 2020a] proposed instead to rely on an expendable projector, a design that is now common practice in SSL [Zbontar et al. 2021, Zhou et al. 2022], and is starting to be adopted by supervised approaches like SupCon [Khosla et al. 2020] and LOOK [Feng et al. 2022]. The impact of these projectors on the representation quality has only seldomly been studied. Wang et al. [2022b] have looked at the impact of projectors, but only for transfer and in isolation. Our work goes one step further and studies how projectors affect performance both on the *training task* and for transfer. We ablate many projector designs and study them jointly with multi-crop. Through our study, we uncover how useful projectors are at navigating the trade-off between training and transfer performance, leading to state-of-the-art results on both dimensions.

4.3 An improved training setup for supervised learning

We now present an improved training setup for learning supervised models that achieve high performance on *both* IN-1K classification and a diverse set of transfer tasks.

Our setup trains a model (or *encoder*) f_θ, parameterized by θ. This model encodes an image I into a transferable representation $x \in \mathbb{R}^d$. We follow the common protocol [Feng et al. 2022, Kornblith et al. 2021] and train all variants of our model on IN-1K using a ResNet50 [He et al. 2016] encoder. This choice of encoder is influenced by recent observations [Wightman et al. 2021] that carefully optimized ResNet50 models perform on par with the best Vision Transformers (ViTs Beyer et al. [2022]) of comparable size on IN-1K. After training our models, we perform transfer learning. We freeze the model's parameters so they are only used to produce transferable representations (x), to be appended with a linear classifier for each transfer task (*e.g.*, IN-1K or any other dataset, see Fig. 4.1b).

Our improved training setup enriches the standard supervised learning paradigm with multi-crop augmentation and an expendable projector head (see Fig. 4.1a). We train our models with one of the two following training objectives: the standard softmax cross entropy loss that learns class weights, or an online variant of nearest class means that is based on class prototypes computed on-the-fly from a memory bank (illustrated in Fig. 4.3). We detail all the proposed improvements below.

Multi-crop data augmentation. Caron et al. [2020] leveraged many image crops of multiple scales and different resolutions when learning invariance to data augmentation in the context of SSL. Their data augmentation pipeline, termed *multi-crop*, is defined over two sets of *global* and *local* crops that respectively retain larger and smaller portions of an image. These crops are processed at different resolutions. We adapt this component to our supervised setup. Given an input image I, we define two scale parameters, for global and local crops, which determine the size ratio between random crops and the image I. We follow Caron et al. [2021] and resize global and local crops to 224×224 and 96×96, respectively. We extract multiple global and local crops, respectively M_g and M_l. Fig. 4.1a illustrates one global $M_g = 1$ and four local $M_l = 4$ crops. In Sec. 4.4, we explore the use of multi-crop for supervised learning, and study the effect of different hyper-parameters under that setting.

Expendable projector head. To countervail the lack of annotations, SSL approaches tackle proxy tasks, such as learning augmentation invariance. In order to prevent the encoder from learning representations that overfit to a potentially unimportant pretext task, SSL architectures often introduce an expendable projector between the encoder and the loss function. On the contrary, for supervised learning, performance on the training task is a major goal in its own right. Here, we aim to learn supervised models that perform well on the training

and on transfer tasks. These two requirements are not aligned and it is necessary to find a trade-off [Kornblith et al. 2021].

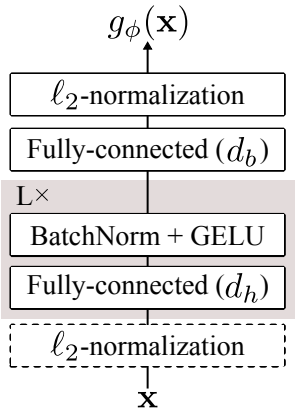


Figure 4.2: Architecture of the projector g_ϕ.

We argue that one can control this trade-off using an additional projector in the context of supervised learning. Similar to SSL methods [Chen et al. 2020a, Chen and He 2021, Chen et al. 2020c] and to the recent SL-MLP [Wang et al. 2022b] we introduce a Multi Layer Perceptron (MLP) projector as part of our supervised training pipeline. Let $g_\phi : \mathbb{R}^d \to \mathbb{R}^{d_b}$ denote this projector, parameterized by ϕ. g_ϕ is composed of an MLP with L hidden layers of d_h dimensions followed by a linear projection to a bottleneck of d_b dimensions. Each hidden layer is composed of a sequence of a linear fully-connected layer, batch-normalization [Ioffe and Szegedy 2015] and a GeLU [Hendrycks and Gimpel 2016] non-linearity. We further apply ℓ_2-normalization to the output of g_ϕ and optionally also to the input. We illustrate this architecture in Fig. 4.2. Note that SL-MLP [Wang et al. 2022b] uses a similar head but with only one hidden layer and no input or output ℓ_2-normalization, so SL-MLP can be seen as a special case of our projector architecture. We compare to their design in Sec. 4.4 and investigate how the number and dimension of hidden layers among other design choices affect the transfer performance of the learned models, verifying and extending the findings of Wang et al. [2022b]. On top of this, we study transfer performance in juxtaposition to performance on the *training task*, and derive the novel insight that projector design allows to control the trade-off between performance on the training task and transferability.

Cosine softmax cross-entropy loss. Incorporating both the components described above in a standard supervised learning paradigm, we can train with the standard softmax cross entropy loss using class labels. The training pipeline is illustrated in Fig. 4.1a. It uses multi-crop data augmentation on each input image I to produce $M = M_g + M_l$ crops I_j, $j = 1,\ldots,M$. Each crop is individually input to the network composed of the encoder followed by the projector, and produces an embedding $z_j = g_\phi(f_\theta(I_j))$. To predict class labels, we multiply embeddings with trainable class weights $W = \{w_c \in \mathbb{R}^{d_b}\}_{c=1}^{C}$, where C is the number of classes. We train the whole pipeline using the *cosine softmax* loss as it was shown to improve IN-1K performance [Kornblith et al. 2021]:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{M}\sum_{j=1}^{M}\sum_{c=1}^{C} y_{[c]} \log \frac{\exp(z_j^\top \bar{w}_c/\tau)}{\sum_{k=1}^{C}\exp(z_j^\top \bar{w}_k/\tau)}, \tag{4.1}$$

where $y \in \{0,1\}^C$ is the C-dim one-hot label vector corresponding to image I, τ is a temperature hyper-parameter and $\bar{w}_c = w_c / \|w_c\|$. Note that projector outputs z are already ℓ_2-normalized.

Online Class Means. Motivated by the recent success of momentum encoders as a way of maintaining online memory banks for large-scale training [He et al. 2020], we revisit the prototype-based Nearest Class Means (NCM) approach of Mensink et al. [2012] and introduce a memory-efficient variant that computes class prototypes in an "online" manner with the help of a memory queue.

Concretely, following Mensink et al. [2012], we define u_c to be the class prototype or *class mean* for class c, *i.e.*, the mean of all embeddings from that class, and define $U = \{u_c\}_{c=1}^C$. Given that we jointly learn class means and the embeddings, computing the exact mean at each iteration is computationally prohibitive. Instead, we formulate an *online* version of NCM that uses a memory bank $\mathcal{Q}$ which stores ℓ_2-normalized embeddings z output by the projector, similar to the memory bank from MoCo [He et al. 2020]. Given the memory $\mathcal{Q}$, we do not learn class weights, but instead compute a *prototype* for each class, on-the-fly, as the average of the embeddings in the memory which belong to that class. Formally, if $\mathcal{Q}_c$ denotes samples in memory that belong to class c, and $N_c = |\mathcal{Q}_c|$, the loss function becomes:

$$\mathcal{L}_{\text{OCM}} = -\frac{1}{M} \sum_{j=1}^{M} \sum_{c=1}^{C} y_{[c]} \log \frac{\exp(z_j^\top \bar{u}_c / \tau)}{\sum_{k=1}^{C} \exp(z_j^\top \bar{u}_k / \tau)}, \tag{4.2}$$

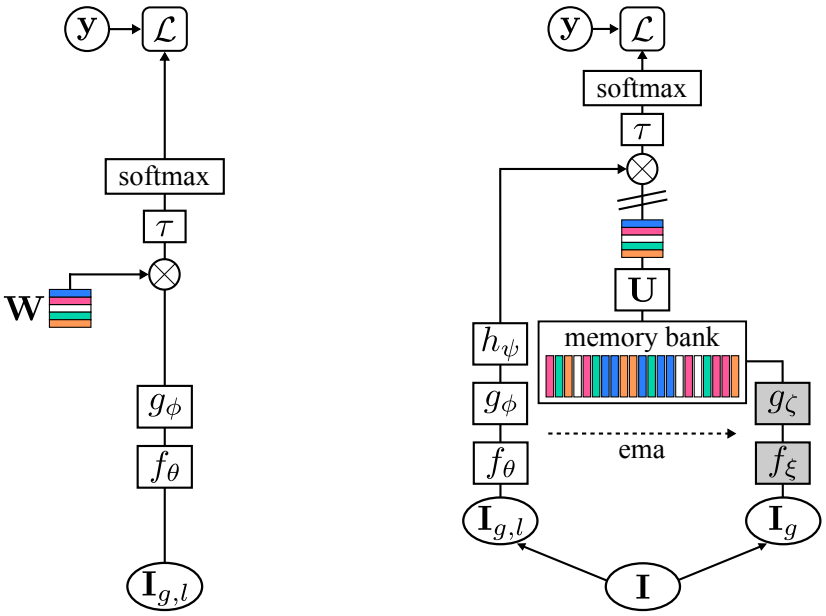
with $\bar{u}_c = \frac{u_c}{\|u_c\|}$ and $u_c = \frac{1}{N_c} \sum_{z \in \mathcal{Q}_c} z$.

We refer to the above training objective as *Online Class Means* or *OCM*. To make sure the embeddings stored in the memory remain relevant as the encoder is updated during training, we follow MoCo [He et al. 2020] and store in memory embeddings from an exponential moving average (EMA) model trailing f_θ and g_ϕ. As we show in our analysis in Sec. 4.4.2, estimating class prototypes using only the relatively small subset of samples in the memory bank leads to class prototypes that drift more across iterations compared to SGD-optimized class weights that converge faster.

An illustration of the model diagrams for Equation (4.1) and Equation (4.2) is given in Fig. 4.3.

4.4 Experiments

In this section, we exhaustively evaluate our proposed training setup on IN-1K and a variety of transfer learning datasets. In Sec. 4.4.1, we study the design of the main components of our setup, *i.e.*, multi-crop augmentation, projectors, and OCM. This leads to a summary of



(a) For Equation (4.1) **(b)** For Equation (4.2), OCM

Figure 4.3: The supervised models we train using our proposed setup. I_g and $I_{g,l}$ represent only global crops or both global and local crops.

our main findings. We then analyze the learned representations, class weights, and prototypes in Sec. 4.4.2. There, we explore how each component affects several facets like feature sparsity, redundancy and variance, average coding length as well as intra-class distance. We also study training dynamics like gradient similarity for multi-crop or how prototypes and classifier weights change across iterations for OCM. In Sec. 4.4.3, we plot the performance of multiple variants of the proposed training setup on the training-versus-transfer performance plane, empirically verifying its superiority over the previous state of the art. Finally, Sec. 4.4.4 presents additional transfer experiments to test the resilience of models to other ImageNet and long-tail datasets.

Protocol. All our models are trained on the training set of ImageNet-1K (IN-1K) [Russakovsky et al. 2015]. Due to the computational cost of training models on IN-1K, each configuration is trained only once. Given an IN-1K-trained model, we discard all the training-specific modules (*e.g.*, the projector g_ϕ, the class weights W), and use the encoder f_θ as a feature extractor, similar to [Kornblith et al. 2019, Sariyildiz et al. 2021]. For each dataset we evaluate on, we learn a linear logistic regression classifier with the pre-extracted features and independently optimize each classifier's hyper-parameters *for every model and every evaluation dataset*.

In all cases, we first extract and store a (single) feature vector for each image and then learn the LogRegclassifiers on top of those features. Our classifiers are therefore trained *without*

data augmentation, and this is why we report lower performance for the RSB model than the one presented by Wightman et al. [2021]. We extract image representations from the encoders f_θ by resizing an image with bicubic interpolation such that its shortest side is 224 pixels and then taking a central crop of size 224×224 pixels. When evaluating on large-scale datasets, such as IN-1K or ImageNet-CoG levels [Sariyildiz et al. 2021], we apply ℓ_2-normalization to the pre-extracted features, and train LogRegclassifiers using SGD with momentum = 0.9 and batch size = 1024 for 100 epochs over a single GPU. When evaluating on smaller-scale datasets (mentioned below) following Kornblith et al. [2019], we train LogRegclassifiers using L-BFGS [Liu and Nocedal 1989]. To this end, we use the implementation in Scikit-learn [Pedregosa et al. 2011]. In both cases, to treat each model as fairly as possible, we set the learning rate and weight decay hyper-parameters for SGD (resp. the inverse regularization coefficient for L-BFGS) using `train`/`val` splits (`val` splits are randomly sampled using 20% of the original `train` splits) and Optuna [Akiba et al. 2019] with at least 25 trials. For each dataset, we repeat this process 5 times with different random seeds and report the average accuracy (variance is negligible). Note that the feature extractor is never fine-tuned, and, because we start from pre-extracted features, no additional data augmentation is used when learning the linear classifiers.[2] This protocol is illustrated in Fig. 4.1b.

Evaluation datasets and measures. We measure performance on the training task by evaluating classification accuracy on the IN-1K validation set. To evaluate transfer learning, we measure classification performance on 13 datasets: the 5 ImageNet-CoG datasets [Sariyildiz et al. 2021] that measure concept generalization, and 8 commonly used smaller-scale datasets: Aircraft [Maji et al. 2013], Cars196 [Krause et al. 2013], DTD [Cimpoi et al. 2014], EuroSAT [Helber et al. 2019], Flowers [Nilsback and Zisserman 2008], Pets [Parkhi et al. 2012], Food101 [Bossard et al. 2014] and SUN397 [Xiao et al. 2010]. We report two metrics: Top-1 accuracy on IN-1K and transfer accuracy via log-odds [Kornblith et al. 2019] averaged over the 13 transfer datasets. Denoting n_{correct} and $n_{\text{incorrect}}$ as the number of correct and incorrect predictions for a dataset, we compute the accuracy p and log odds score as follows:

$$p = \frac{n_{\text{correct}}}{n_{\text{correct}} + n_{\text{incorrect}}}, \quad \text{log odds} = \log \frac{p}{1-p}. \tag{4.3}$$

Note that we provide per dataset results in Tab. D.1 of the Appendix. Sec. 4.4.4 present additional evaluations on IN-1K-Sketch [Wang et al. 2019], IN-1K-v2 [Recht et al. 2019] IN-1K-R [Hendrycks et al. 2021a], IN-1K-A [Hendrycks et al. 2021b] and two long-tail datasets: i-Naturalist 2018 and 2019 [Van Horn et al. 2018].

[2] Although in their evaluation Caron et al. [2021], Zhai et al. [2019a] train linear classifiers with data augmentation or fine-tune the encoder while training classifiers, we found that such protocols make a proper hyper-parameter validation computationally prohibitive. We instead follow the linear evaluation protocol from Kornblith et al. [2019] and Sariyildiz et al. [2021].

Figure 4.4: Impact of the number of local crops (M_l) on the performance on IN-1K (*left*) and transfer datasets (*right*) when varying the number of hidden layers (L) in the projector. The number of global crops (M_g) is 1 in all cases.

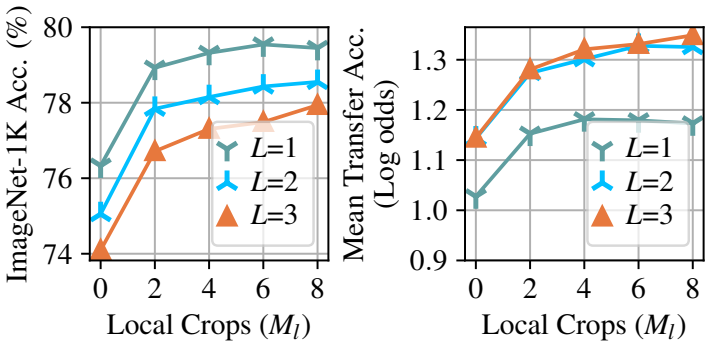


Table 4.1: Impact of the projector size on performance, via the number of hidden layers L (*left*) and hidden units d_h (*right*). The default configuration: L=1, d_h=2048, d_b=256 and with ℓ_2-normalization of the input (highlighted rows). We use $M_g = 1$ and $M_l = 8$ ("Base+Mc").

	IN1K	Transfer
Base	76.6	0.10
Base+Mc	79.7	0.25
$L = 1$	<u>79.8</u> ↑	1.15
$L = 2$	78.6	1.31
$L = 3$	77.5	<u>1.33</u> ↓

d_h	IN1K	Transfer
512	<u>80.0</u> ↑	0.82
1024	<u>80.0</u>	1.06
2048	79.8	1.15
4096	79.8	1.20
8192	79.4	<u>1.22</u> ↓

Implementation details. f_θ is a ResNet50 [He et al. 2016] encoder, trained for 100 epochs with mixed precision in PyTorch [Paszke et al. 2019] using 4 GPUs where batch norm layers are synchronized. We use an SGD optimizer with 0.9 momentum, a batch size of 256, 1e-4 weight decay and a learning rate of $0.1 \times \text{batch size}/256$, which is linearly increased during the first 10 epochs and then decayed with a cosine schedule. We set $\tau = 0.1$ and, unless otherwise stated, we use the data augmentation pipeline from DINO [Caron et al. 2021] with 1 global and 8 local crops (M_g = 1 and M_l = 8). Training one of our models takes up to 3 days with 4 V100 GPUs depending on its projector configuration.

4.4.1 Analysis of component design and hyper-parameters

Multi-crop data augmentation. We first study the effect of the number of local crops on IN-1K and transfer performance. We train supervised models using Equation (4.1) with 1 global and 2, 4, 6 or 8 local crops, and projectors composed of 1, 2 or 3 hidden layers, and report results in Fig. 4.4. Our main observations are: a) training with local crops improves the performance on both IN-1K and transfer tasks, and b) although increasing the number

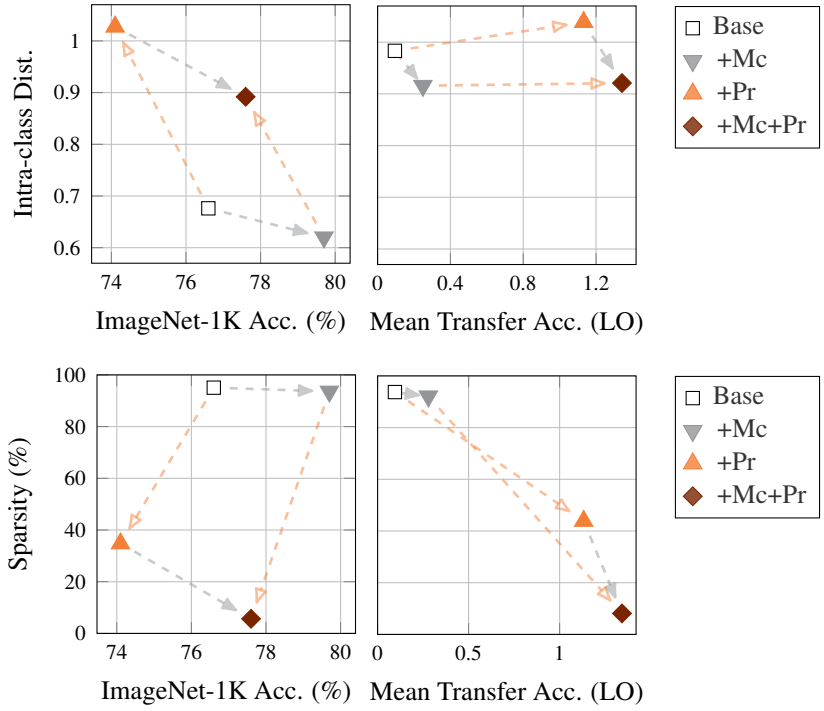


Figure 4.5: Average intra-class ℓ_2-distance between samples from the same class (*top*) and sparsity as the percentage of feature dimensions close to zero (*bottom*), on IN-1K and averaged over transfer datasets. Gray and Orange arrows denote changes due to adding multi-crop and projectors, respectively. Best viewed in color.

of local crops generally helps, performance saturates with 8 local crops. We set $M_g = 1$ and $M_l = 8$ for all subsequent evaluations.

Note that using local crops increases the effective batch size, which, in turn, increases training time. We therefore conduct two experiments to see if a longer training or a larger batch size would lead to similar gains. We train two models using a single crop sampled from a wide scale range (*i.e.*, able to focus on both large and small image regions), one with 9×larger batch size, the other for 800 epochs. Unlike multi-crop, these models bring no significant gain.

Expendable projector head. We study the impact of different architectural choices and hyper-parameters for the projector. We vary the number of hidden layers (L), the dimension of the hidden (d_h) and bottleneck (d_b) layers, and whether or not to ℓ_2-normalize the projector input (ℓ_2). We start from a default configuration: $L = 1$, $d_h = 2048$, $d_b = 256$ and with ℓ_2-normalized inputs. We ablate each parameter separately by training models optimizing Equation (4.1), *i.e.*, without the OCM component. We use multi-crop in all cases.

The most interesting results from this analysis are presented in Tab. 4.1. We see that *the number of hidden layers (L) is an important hyper-parameter that controls the trade-off between*

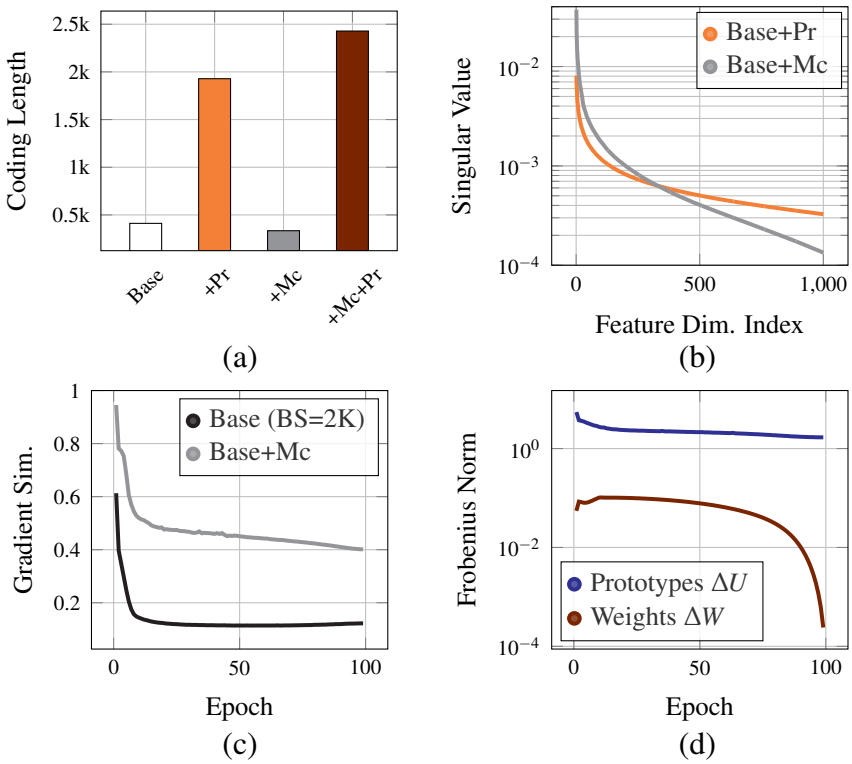


Figure 4.6: (a) Average coding length per sample [Yu et al. 2020] over all *transfer* datasets. (b) Singular values across dimensions, averaged over the transfer datasets. We show the first 1000 dimensions (of 2048) for clarity. (c) Average similarity between class weight gradients $\nabla_{w_c}\mathcal{L}_{\text{CE}}$ during training. (d) Change in class weights W and prototypes U at every iteration across all classes (see text for details) for models trained using Equation (4.1) and Equation (4.2), respectively. Best viewed in color.

IN-1K and transfer performance. Adding a projector head with a single hidden layer not only improves the already strong IN-1K performance of multi-crop (Base+Mc in Tab. 4.1), but also significantly boosts its average transfer performance. More hidden layers seem to increase transfer performance, at the cost of a decrease in IN-1K accuracy. The same can be said about the dimension of the hidden layer, yet we further see that a larger d_h significantly increases transfer performance, and moderately decreases IN-1K accuracy. On the contrary, we observe that the bottleneck dimension d_b and input ℓ_2-normalization only have a small influence on IN-1K and transfer performance. Overall, our observations verify and significantly extend the ones recently presented by Wang et al. [2022b]. We not only study the design of projectors jointly with multi-crop, but also analyse transfer performance jointly with performance on IN-1K, revealing a *trade-off* between the two, that is fully controlled through the design of the project head.

Online class means. There are two main hyper-parameters in OCM: the size of the memory bank and the momentum of the EMA models that populate the memory bank and provide the

embeddings for class prototypes. We explored momentum values 0 and 0.999 (in the former, we directly use f_θ and g_ϕ to compute prototypes) and see that trailing EMA is essential for maintaining high performance, *i.e.*, momentum $= 0$ performs poorly, aligning with the observations for MoCo [He et al. 2020]. However, unlike MoCo or other recent methods such as LOOK [Feng et al. 2022], OCM does not require a large memory bank to achieve the highest performance. We experimented with memory sizes between 2048 and 65546, and found that 8192 works best.

4.4.2 Analysis of learned features, class weights and prototypes

We now investigate how different components of our setup affect training or the learned representations. We analyse the features produced, class weights and prototypes from the following models: a) *Base*: a model trained using cosine softmax loss without multi-crop and projector, b) *Base (BS=2K)*: Base but with 9×larger batch size, c) *Base+Mc*: Base with multi-crop, d) *Base+Pr*: Base with a projector, e) *Base+Mc+Pr*, and f) *OCM*: a model trained using Equation (4.2).

Intra-class distance. We start by analysing the ℓ_2-normalized features for the four models, Base, Base+Mc, Base+Pr and Base+Mc+Pr, by computing the average ℓ_2-distance between samples from the same class (*i.e.*, intra-class distance). We see in Fig. 4.5 (left) that multi-crop reduces intra-class distance on IN-1K, while projectors increase it. Not surprisingly, this correlates with training task performance, *i.e.*, lower intra-class distance translates to better performance on IN-1K. On the transfer datasets, however, we found no strong correlation between the two, *i.e.*, transfer performance does not necessarily depend on intra-class distance.

Sparsity. In Fig. 4.5 (right) we report feature *sparsity ratio*, *i.e.*, the percentage of feature dimensions close to zero for ℓ_2-normalized features from the four models. We see that: a) the average sparsity ratio on the transfer datasets is inversely correlated with performance, *i.e.*, linear classifiers trained on *less sparse features achieve better transfer performance*, and b) *projectors dramatically reduce sparsity*. We find this last observation intuitive: features from the layer right before the cross-entropy loss are encouraged to be as close to a one-hot vector as possible and therefore sparse. Introducing projectors in between allows the encoder to output less sparse features, which improves transfer.

Coding length. To further investigate our observations on sparsity, we follow Yu et al. [2020] and compute the average coding length per sample on the transfer datasets (see Fig. 4.6 (a)). We see that projectors largely increase the "information content" of representations. This was also verified by analysing singular values per dimension for models with and without

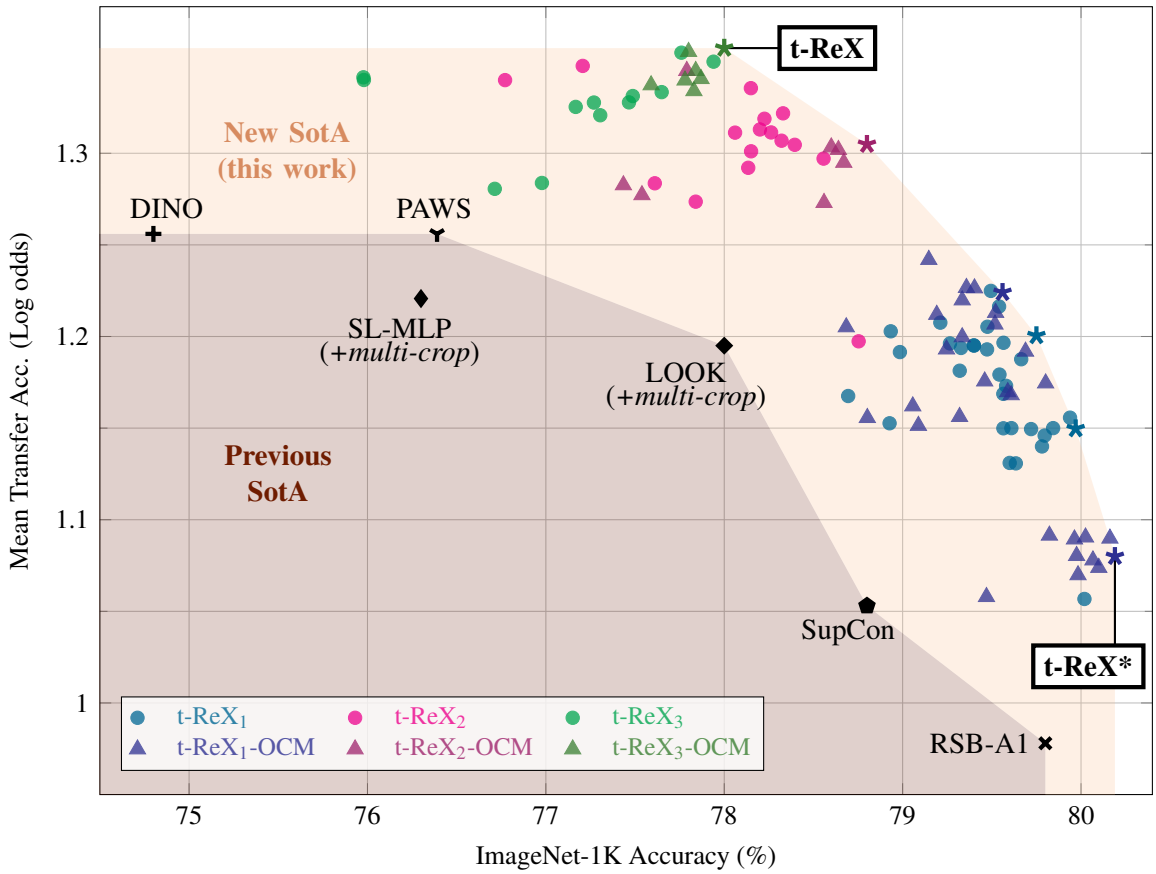


Figure 4.7: Comparison on the training task vs transfer task performance for ResNet50. We report IN-1K (Top-1 accuracy) and transfer performance (log odds) averaged over 13 datasets (5 ImageNet-CoG levels, Aircraft, Cars196, DTD, EuroSAT, Flowers, Pets, Food101 and SUN397) for a large number of our models trained with the supervised training setup presented in Sec. 4.3. Models on the convex hull are denoted by stars. We compare to the following state-of-the-art (SotA) models: Supervised: RSB-A1 [Wightman et al. 2021], SupCon [Khosla et al. 2020], SL-MLP [Wang et al. 2022a] and LOOK [Feng et al. 2022] with multi-crop; self-supervised: DINO [Caron et al. 2021]; semi-supervised: PAWS [Assran et al. 2021].

projectors (Base+Pr and Base+Mc). For each model, we compute singular values on each transfer dataset which are normalized by their sum so that they sum to 1. We then sort these normalized singular values by decreasing order, and average them over transfer datasets. As can be seen in Fig. 4.6 (b), feature variance is more uniformly distributed over dimensions when a projector is used. These observations might explain why projectors reduce overfitting to IN-1K concepts.

Gradient similarity. To understand why using multi-crop increases performance for the same batch size, we examine the gradients of class weights $\nabla_W \mathcal{L}_{\text{CE}}$ for two models that

have the same effective batch size, with and without multi-crop. At each training iteration, we compute the average cosine similarity between individual gradients of every pair of class weights $\nabla_{W_{c_i}} \mathcal{L}_{\text{CE}}$ and $\nabla_{W_{c_j}} \mathcal{L}_{\text{CE}}$ for any $c_i \neq c_j$. As we see from Fig. 4.6 (c), cosine similarity increases substantially with multi-crop. In other words, on average, classifier gradients (and therefore the class weights themselves) are more entangled. We attribute this to the fact that some of the local crops (*e.g.*, the ones that mostly cover background and hence are not really discriminative for the class at hand) are harder to classify.

Change in class weights and prototypes. To understand the differences between the training objectives in Equation (4.1) and Equation (4.2), we measure how much class weights W and prototypes U change during the training phase. In Fig. 4.6 (d), we plot the average change over all classes by computing the Frobenius norm between before and after each iteration, *i.e.*, $\Delta W = \|\bar{W}^t - \bar{W}^{t-1}\|_2$ and $\Delta U = \|\bar{U}^t - \bar{U}^{t-1}\|_2$, where t is the training iteration, and $\bar{W}$ and $\bar{U}$ are the class weight w_c and prototype u_c ℓ_2-normalized per class and concatenated, respectively. Interestingly, we observe that *prototypes U change orders of magnitude more than class weights W throughout training*. We believe this is because we compute class prototypes using only the small subset of images from our memory bank. The average number of samples per class on IN-1K is 1281, whereas, on average we have only 8 per class in the memory bank. We argue that this prevents OCM from overfitting, leading to higher IN-1K performance, as we show next.

4.4.3 Pushing the envelope of training-versus-transfer performance

In this section we report and analyse results from more than 100 different models trained on IN-1K, all different instantiations of the proposed training setup. We varied hyper-parameters such as the number of hidden layers in the expandable projector head or the training objective. The most important results are depicted in Fig. 4.7.

Previous state of the art. RSB-A1 [Wightman et al. 2021] is a highly optimized supervised ResNet50 model with top performance on IN-1K. The self-supervised DINO model [Caron et al. 2021] has shown top transfer learning performance, while also performing well on IN-1K. The semi-supervised PAWS [Assran et al. 2021] model matches DINO in transfer performance, with improved IN-1K accuracy. To our knowledge, RSB-A1 and DINO/PAWS are the current state-of-the-art ResNet50 models for IN-1K classification and transfer learning respectively. We also compare to three recent supervised models: SupCon [Khosla et al. 2020], LOOK [Feng et al. 2022] and SL-MLP [Wang et al. 2022a]. For all models except LOOK and SL-MLP, we evaluate the models provided by the authors. Due to the absence of official code we reproduced LOOK and SL-MLP ourselves, enhancing them with multi-crop.

Our reproductions achieve higher performance than the one reported in the original papers. In both cases, we use a projector with 1 hidden layer.

Notations. Models trained with the basic version of the proposed training setup, *i.e.*, using multi-crop, a projector with L hidden layers and a cosine softmax cross-entropy loss are reported as **t-ReX**$_L$. For models using the OCM training objective we append **-OCM**. Models on the "envelope" (*i.e.*, the convex hull) of Fig. 4.7 are highlighted with a star (exact configurations are in the Appendix: Tabs. C.1 and D.1).

Main results. Our main observations from results presented in Fig. 4.7 can be summarized as follows.

- ***Pushing the envelope.*** Many variants from our supervised training setup "push" the envelope beyond the previous state of the art, across both axes. Several of these models improve over the state of the art on one or the other axis, but no single model outperforms all the others on both dimensions. As the number of hidden layers of the projector increases, models gradually move from the lower right to the upper left corner of the plane. This shows again that increasing the projector complexity improves transfer performance at the cost of IN-1K (training task) performance.
- ***No reason for no supervision.*** A large number of supervised variants outperform the DINO method with respect to transfer learning, while also being significantly better on IN-1K. We therefore show that training with label supervision does not necessarily require to sacrifice transfer learning performance and one should use label information if available.
- ***State-of-the-art IN-1K performance with three simple modifications.*** A number of t-ReX$_1$ models outperform the highly optimized RSB-A1 on IN-1K, essentially by using only three components over the "vanilla" supervised learning process that is considered standard practice: a) cosine softmax with temperature, b) multi-crop data augmentation, and c) an expendable projector.
- ***Training with class prototypes brings further gain.*** Given the same projector configuration, training models with the OCM objective (Equation (4.2)) has a small advantage over training with cosine softmax (Equation (4.1)). We see that 4 of the 6 points on the convex hull in Fig. 4.7 are t-ReX-OCM models. This suggests that using class prototypes is a viable alternative to learning class weights end-to-end.
- ***Introducing t-ReX and t-ReX*.*** We single out the two instantiations that respectively excel on the transfer learning and IN-1K axes, *i.e.*, **t-ReX**$_3$**-OCM** and **t-ReX**$_1$**-OCM**. We rename them **t-ReX** and **t-ReX***, respectively. We envision these two **t**ransferable **Re**sNet50 models and their corresponding training setups to serve as strong supervised

Table 4.2: Results on IN-1K concepts. For each model, we report results on the IN-1K "Val" set (the x-axis of Fig. 4.7), as well as on the test sets of IN-1K-v2 [Recht et al. 2019], IN-1K-sketch [Wang et al. 2019], IN-1K-R [Hendrycks et al. 2021a] and IN-1K-A [Hendrycks et al. 2021b], using in all cases the encoder and the linear classifier trained on the original IN-1K training set. IN-1K-v2 numbers are averaged over the three test sets (matched-frequency, threshold-0.7 and top-images).

Model	Val	v2	Sketch	R	A
DINO	74.8	69.9	19.8	31.9	4.9
PAWS	76.4	71.6	24.2	37.1	5.2
SupCon	78.8	74.3	**30.9**	41.3	9.7
RSB-A1	79.8	75.4	27.9	38.9	7.9
t-ReX	78.0	73.6	26.8	39.1	7.0
t-ReX*	**80.2**	**76.2**	29.1	**41.8**	**11.7**

baselines for future research on transfer learning and IN-1K. All the hyper-parameters for these two models are in Tab. C.1 in the Appendix.

4.4.4 Evaluations on the additional transfer datasets

In our experiments so far, we have measured transferability of representations on 13 datasets (5 ImageNet-CoG levels and 8 other smaller-scale transfer datasets). Here we extend our transfer evaluations to 6 more datasets, *i.e.*, 4 ImageNet and 2 long-tail datasets.

ImageNet datasets. We compare **t-ReX** and **t-ReX*** to the previous state of the art on the four ImageNet datasets, namely IN-1K-sketch [Wang et al. 2019], IN-1K-v2 [Recht et al. 2019], IN-1K-R [Hendrycks et al. 2021a] and IN-1K-A [Hendrycks et al. 2021b]. As before, for each model, we use the trained encoder as a feature extractor. We reuse the linear classifier trained on IN-1K and apply it directly to the test images of these four ImageNet datasets. Note that there are 3 test sets for IN-1K-v2, and we evaluate over all of them and report their average. Tab. 4.2 presents our results. Looking at the mean Top-1 accuracy over the three test sets of IN-1K-v2, we observe that **t-ReX*** also matches the performance of RSB-A1, outperforming all others. On the other hand, SupCon performs the best on IN-1K-Sketch, where **t-ReX*** is the second best. We think that the contrastive loss used in SupCon might have improved its out-of-distribution robustness for the training concepts. On IN-1K-R and IN-1K-A, **t-ReX*** is superior than all the other models. Overall, we see that **t-ReX*** shows strong generalization capabilities to all four ImageNet datasets.

Long tail datasets. We evaluate the long-tail transfer classification performance of DINO, PAWS, RSB-A1, **t-ReX** and **t-ReX*** on two class-imbalanced datasets, iNaturalist 2018

Table 4.3: Transfer results on long-tail classification. For each model, we train linear classifiers on the iNaturalist 2018 and iNaturalist 2019 datasets [Van Horn et al. 2018] with class-imbalanced data, following the LogReg protocol from ImageNet-CoG.

Model	iNaturalist 2018	iNaturalist 2019
DINO	41.9	51.4
PAWS	40.8	49.8
RSB-A1	34.9	43.2
t-ReX	**45.8**	**54.2**
t-ReX*	36.0	44.2

and iNaturalist 2019 [Van Horn et al. 2018]. For these evaluations, we follow the LogReg protocol from the ImageNet-CoG benchmark [Sariyildiz et al. 2021]. Results are reported in Tab. 4.3. We see that our **t-ReX** and **t-ReX*** models still outperform RSB-A1 and DINO respectively, despite a challenging long-tail class distribution.

4.5 Conclusion

We present a supervised training setup that leverages components from self-supervised learning, and improves generalization without conceding on the performance of the original task, *i.e.* IN-1K classification. We also show that substituting class weights with prototypes used an online class mean classifier over a small memory bank boosts performance even further. We extensively analyze the design choices and parameters of those models, and show that many variants push the envelope on the IN-1K-transfer performance plane. This validates our intuition that image-level supervision, if available, can be beneficial to both IN-1K classification and transfer tasks.

Chapter 5

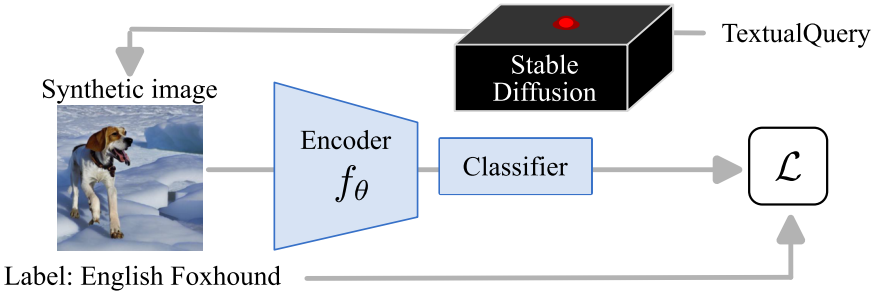
Learning transferable representations from synthetic ImageNet clones

Contents

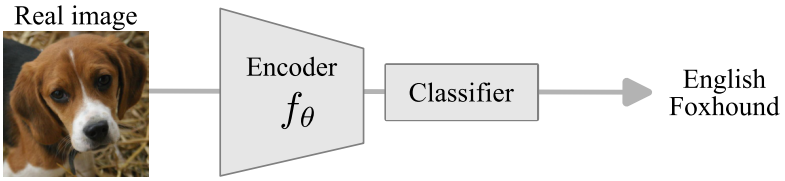
5.1 Introduction . . . 79
5.2 Related work . . . 81
5.2.1 Learning with synthetic data . . . 81
5.2.2 Distillation of datasets and models . . . 82
5.3 Preliminaries . . . 83
5.4 Generating synthetic ImageNet clones . . . 84
5.4.1 Generating datasets using class names . . . 84
5.4.2 Addressing issues with semantics and domain . . . 86
5.4.3 Increasing the diversity of generated images . . . 86
5.5 Experiments . . . 88
5.5.1 Results on ImageNet datasets . . . 89
5.5.2 Resilience to domain shifts . . . 91
5.5.3 Transfer learning . . . 92
5.5.4 Impact of guidance scale and diffusion steps . . . 94
5.5.5 Analysis of the learned features . . . 94
5.6 Discussion . . . 97
5.7 Conclusions . . . 98

The previous chapter focused on the model aspect of learning visual representations, and presented a supervised learning model that learns transferable representations from the ImageNet-1K dataset [Russakovsky et al. 2015]. This chapter shifts its focus to the data aspect, and explores whether synthetic images (more specifically, synthetic ImageNet-1K clones) can be used to learn transferable representations. The work presented in this chapter[1] is accepted at

[1]Project page: https://europe.naverlabs.com/imagenet-sd



(a) Training a model on synthetic images.



(b) Testing the frozen model on real images.

Figure 5.1: Overview of our experimental protocol. During training, the model has access to synthetic images generated by the Stable Diffusion model, provided with a set of prompts per class. During evaluation, real images are classified by the frozen model.

IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2023 [Sariyildiz et al. 2023a].

The main motivation behind this chapter is the observation that recent image generation models such as Stable Diffusion [Rombach et al. 2022] exhibits an impressive ability to generate fairly realistic images starting from a simple text prompt. Then we ask: Could such models render real images obsolete for training image prediction models? We answer part of this provocative question by investigating the need for real images when training models for ImageNet classification. Provided only with the class names that have been used to build the dataset, we explore the ability of Stable Diffusion to generate synthetic clones of ImageNet and measure how useful these are for training classification models from scratch. We show that with minimal and class-agnostic prompt engineering, ImageNet clones are able to close a large part of the gap between models produced by synthetic images and models trained with real images, for the several standard classification benchmarks that we consider in this study. More importantly, we show that models trained on synthetic images exhibit strong generalization properties and perform on par with models trained on real data for transfer.

5.1 Introduction

The rise of (shallow) machine learning [Chen et al. 2001, Vedaldi et al. 2009] and later deep learning [He et al. 2016, Krizhevsky et al. 2012, Szegedy et al. 2015] has entirely changed the landscape of computer vision research over the past few decades, shifting some of the focus from *methods* to the *training data* itself. Datasets, initially of hundreds of images and dozens of classes [Everingham et al. 2009, Fei-Fei et al. 2004], have grown in size and complexity, and started becoming contributions in their own right. They have been fueling the progress of computer vision as much as, if not more than, the methods themselves. ImageNet [Deng et al. 2009], and mainly its ImageNet-1K [Russakovsky et al. 2015] subset of about 1 million annotated images, has impacted the field in an unprecedented way. Yet, curating and annotating such a dataset comes at a high money and labor cost.

The last couple of years have seen the rise of large and generic models, trained on data which is less curated but orders of magnitude larger. Those proved to be easily applicable, either directly, or combined with a tailored model, to a wide range of computer vision transfer tasks [Ilharco et al. 2021, Jia et al. 2021, Radford et al. 2021b]. They have also been used beyond prediction tasks, *e.g.*, for text-conditioned image generation. Models such as DALL-E [Ramesh et al. 2021] or Stable Diffusion [Rombach et al. 2022] have demonstrated impressive image generation ability. They produce fairly realistic synthetic images and exhibit a high degree of compositionality.

Such generative models are trained on billion-scale datasets [Schuhmann et al. 2022] composed of noisy image-text pairs scraped from the internet. Although training such models is out of reach for most institutions, a few of them have been made available to the community. Given the remarkable ability of these generative models, it is only natural to ask provocative questions such as: *Is there still a need for real images when training image prediction models?*

In this chapter, we explore this question through one of the most iconic computer vision datasets, ImageNet [Deng et al. 2009]. We study to which extent this dataset can be entirely replaced by synthetic images when learning deep models. For this, we assume that we are provided with a set of classes, and the Stable Diffusion [Rombach et al. 2022] model a generator that can produce realistic images from a textual prompt.

Our task is to learn an image classification model *from scratch* using a dataset composed only of synthetic images. We then evaluate the performance of this model on several datasets. (These two phases are illustrated in Fig. 5.1a and Fig. 5.1b, respectively.) First and foremost, we measure how well models and classifiers trained only on synthetic images recognize the training classes in real images from the standard ImageNet validation set. Then, we evaluate them on common datasets that test their resilience to domain shifts or adversarial examples,

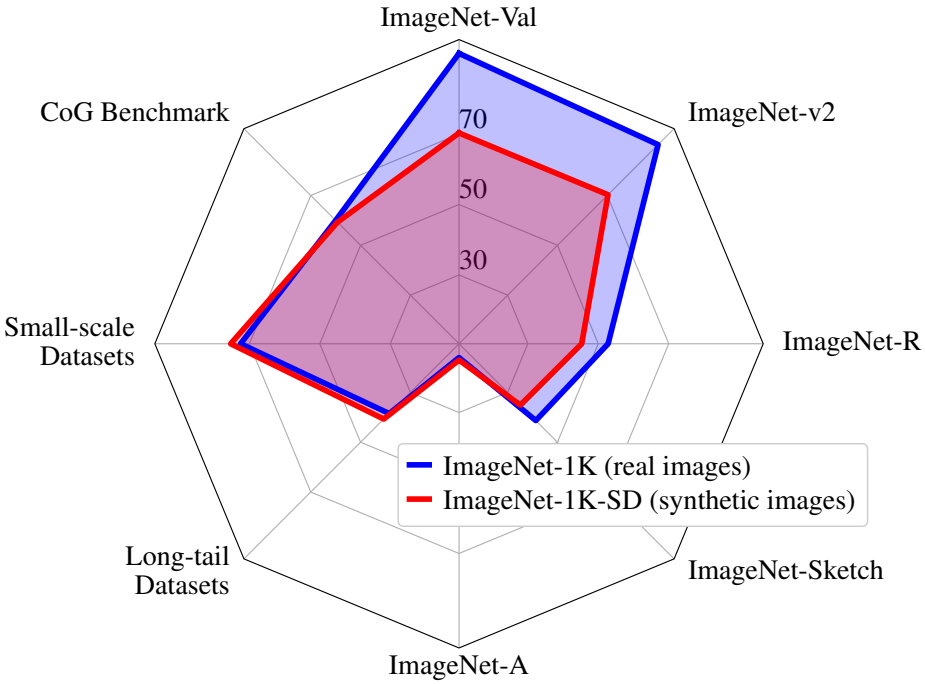


Figure 5.2: ImageNet-1K vs ImageNet-1K-SD. The blue polygon shows the performance of a model trained on ImageNet-1K. The red polygon depicts the performance of one trained on ImageNet-1K-SD, *i.e.*, only on synthetic data generated with Stable Diffusion [Rombach et al. 2022] using the class names of ImageNet-1K. We report Top-5 accuracy for ImageNet test sets, and average Top-1 for transfer tasks.

still for the ImageNet training classes. Finally, we consider several transfer learning scenarios where we measure the generalization performance of our models to novel classes. Fig. 5.2 summarizes the main results by comparing models trained on two equally sized set of images from the same set of classes, one real and one synthetic, on a number of these tasks. The gap is surprisingly narrow, especially for some of these scenarios.

To summarize, our contributions in this chapter are threefold:

- First, we leverage Stable Diffusion [Rombach et al. 2022] and generate synthetic ImageNet clones, *i.e.*, datasets with synthetic images for the ImageNet classes, using class names as prompts. We analyse the generated images, highlight important issues, and propose class-agnostic alterations to the basic prompt that reduce semantic issues and increase diversity.
- Second, we train classification models using different ImageNet clones and show that they can achieve 91.7% and 70.3% Top-5 accuracy on ImageNet-100 and ImageNet-1K respectively.

- Finally, we evaluate the generalization capacity of our models. We show that their performance gap with models trained on real images is reduced when testing for resilience to domain shifts or adversarial examples. Moreover, we show that our models perform on par with models trained conventionally when testing on 15 transfer datasets.

5.2 Related work

5.2.1 Learning with synthetic data

Learning with synthetic data has become a standard way to create large amounts of labeled data for annotation heavy tasks, such as human understanding [Pumarola et al. 2019, Varol et al. 2017], semantic segmentation [Chen et al. 2019b, Sankaranarayanan et al. 2018], optical flow estimation [Dosovitskiy et al. 2015, whan Kim et al. 2022] or dense visual alignment [Peebles et al. 2022]. In most cases, this synthetic data requires access to 3D models and renderers [Mahmood et al. 2019], or to a simulator [Richter et al. 2016] with a physically plausible engine. Recent works propose pretraining on a database of synthetic fractal [Kataoka et al. 2022] or sinusoidal wave [Takashima et al. 2023] images before fine-tuning the model using real images on a downstream task. In this study we use synthetic data to learn encoders and classifiers that can be used *out-of-the-box*, without the need for a subsequent fine-tuning step. Closest to our work, Kumar et al. [2022] generate synthetic OCT images to train a glaucoma detection model to be applied to real images. Here, we target synthetic clones of complex natural image datasets, *i.e.*, ImageNet-1K [Russakovsky et al. 2015], and we use a *general-purpose* text-to-image generation model.

Synthetic ImageNet clones. Synthetic images for ImageNet classes have been used recently in a number of related works [Besnier et al. 2020, Li et al. 2022, Ravuri and Vinyals 2019] based on class conditional Generative Adversarial Networks (GANs), such as BigGAN [Brock et al. 2019]. Besnier et al. [2020] generate images for ten ImageNet classes and propose techniques to reduce the gap between models trained on generated images and real ones. Li et al. [2022] synthesize five images for each ImageNet-1K class, together with their semantic segmentation annotations to automatically generate pixel-level labels at scale. Our work focuses on image-level classification, and uses a general-purpose text-conditioned generative model instead of ImageNet-1K class-conditioned GANs. It further offers a larger scale study with promising results on the full ImageNet-1K benchmark when training from 1.28 million synthetic images. Concurrent work [He et al. 2023] also synthesizes data for ImageNet-1K, but focuses on improvements on top of the CLIP [Radford et al. 2021b] model or after fine-tuning.

Synthetic images as data++. Data sampled from generative models [Goodfellow et al. 2020, Ho et al. 2020, Ramesh et al. 2021, Rombach et al. 2022] can be seen as data with added functionalities or "data++" [Isola 2022]. Such data can be manipulated, interpolated or composed [Chai et al. 2021a,b, Jahanian et al. 2020, 2022] with dedicated operators in their latent space, and further used for counterfactual reasoning [Liu et al. 2019, Mao et al. 2021, Oktay et al. 2018]. In this work, we do not exploit these added functionalities. Our prompts consider a class at a time and do not leverage any interpolation nor the composition properties of synthetic data. Instead, we chose our complete pipeline, including the set of data augmentations, to be identical to the one we use for real images, to allow for a fair comparison.

Zero-shot learning and test-time view synthesis. Generative models have been used to extend models to new classes, or to create novel views at test time. Chai et al. [2021b] synthesize novel views for test-time ensembling by perturbing the latent code of a test image. Aiming at zero-shot recognition [Xian et al. 2018b], Elhoseiny et al. [2013] synthesize a classifier for any novel class given its semantic description (*e.g.*, textual or attribute-based), whereas others synthesize images [Dunlap et al. 2023, Gu et al. 2022], or image *features* [Lazarou et al. 2022, Sariyildiz and Cinbis 2019] using such descriptions. Here we aim to learn encoders from scratch, and do not rely on models previously trained on real data.

5.2.2 Distillation of datasets and models

Knowledge distillation [Buciluǎ et al. 2006, Hinton et al. 2014] is a mechanism to transfer knowledge from a pretrained "teacher" model into a "student" one, and it usually requires images. Our approach can be seen as performing *image-free* distillation from a generic text-to-image generation model into a specific classification model. We assume no access to images to distill from and, instead of distilling the visual encoder of the image generation model, inspired by recent works in NLP [Ma et al. 2022], we prompt a generation model to produce synthetic images and train a classifier with them.

Dataset distillation [Cazenavette et al. 2022, Zhao et al. 2021a], on the other hand, is a way of compressing a training set of real images into a smaller set of synthetic images such that after training a model on those, it performs as well as if it had been trained on the original set. However, one needs to tailor the generation process to a specific task, whereas in our case, we sample images from a task-agnostic generator.

Reconstructing images from model activations can be considered as another form of distillation. Earlier works reconstruct images from gradient-based features [Vondrick et al. 2013, Weinzaepfel et al. 2011] or CNN activations [Mahendran and Vedaldi 2015]. Since

then many methods have tried to uncover the training data distribution as it is stored in the weights of a model [Chen et al. 2019a, Yin et al. 2020]. Instead of trying to recover the training distribution of the teacher image generation model, we use prompting to distill its knowledge for a specific image classification task.

5.3 Preliminaries

In this section, we first define the task we solve, *i.e.*, learning an image classification model when the training set of real images is replaced by an image generator, and training proceeds using only synthetically generated images. We then briefly describe Stable Diffusion [Rombach et al. 2022], *i.e.*, the text-to-image generation model we use in this work.

Task formulation. Our goal is to learn an image classification model given a set of class names $\mathcal{C}$ and a text-to-image generator $\mathcal{G}$. This task is a variant of image classification where the fixed-size image training set is replaced by an image generator. The model we aim to learn consists of an encoder $x = f_\theta(I)$ that maps an *image* I into a vector representation $x \in \mathbb{R}^d$, and a classifier $o = Wx$ that outputs a class prediction distribution $o \in \mathbb{R}^{|\mathcal{C}|}$ over $|\mathcal{C}|$ classes $c_i \in \mathcal{C}$, where $W \in \mathbb{R}^{|\mathcal{C}| \times d}$, $i = \{1, .., |\mathcal{C}|\}$ and $|.|$ denotes the cardinality of a set. We follow the common supervised learning setting [Krizhevsky et al. 2012, Russakovsky et al. 2015] and, unless otherwise stated, learn the encoder parameters θ together with the classifier parameters W for the task. This model (encoder and classifier) is evaluated on the initial classification task, by applying it to real images (Sec. 5.5.1 and Sec. 5.5.2). We also evaluate the visual encoder in the context of several transfer learning tasks (Sec. 5.5.3).

Text-to-image with Stable Diffusion. We use the recent Stable Diffusion model [Rombach et al. 2022] (SD) as text-to-image generator $\mathcal{G}$. SD is a denoising diffusion model [Ho et al. 2020] built around the idea of *latent diffusion*. The diffusion process is run on a compressed latent space for efficiency. An image encoder/decoder is used to interface the latent diffusion model with the pixel space. The generation process can be conditioned in many ways, *e.g.*, with text for text-to-image generation, or an image latent vector for image manipulation.

The text-to-image SD model consists of three main components: **a)** an autoencoder whose visual encoder outputs a structured latent representation that is fed as input to the forward diffusion process and whose decoder is then used to convert the latent vectors back to pixels, **b)** a denoising U-Net that runs the diffusion process, and **c)** a text encoder, *i.e.*, similar to the one used by CLIP [Radford et al. 2021b].

The text-to-image generation process takes a textual prompt p as input and generates an image $I \in \mathbb{R}^{w \times h \times 3}$. Let $g(p)$ denote the generation function of model $\mathcal{G}$. Image I is then given by $I = g(p)$. In practice, the prompt p is first encoded via the text encoder and the

text embedding is used as a conditioning vector for the latent diffusion process that runs for a number of steps. The latent representation is then provided to the decoder, which outputs the image I.

There are two important parameters that control the quality and speed of text-conditioned diffusion; the number of diffusion steps and the coefficient that weights the textual conditioning vector. The former is linearly related to extraction time, while the latter provides an excellent way of controlling the visual diversity of generated images. The default values are 50 steps and guidance scale equal to 7.5.

Link to distillation. Since the generator is a model that internally encodes visual information, the image classification model we learn is essentially derived from $\mathcal{G}$. Under this formulation, and as discussed in Sec. 5.2, one can also see this task as text-guided, image-free knowledge distillation. Here we distill knowledge from a model of a very different nature, *i.e.*, a text-to-image generation model, to a purely visual encoder, for solving a specific task.

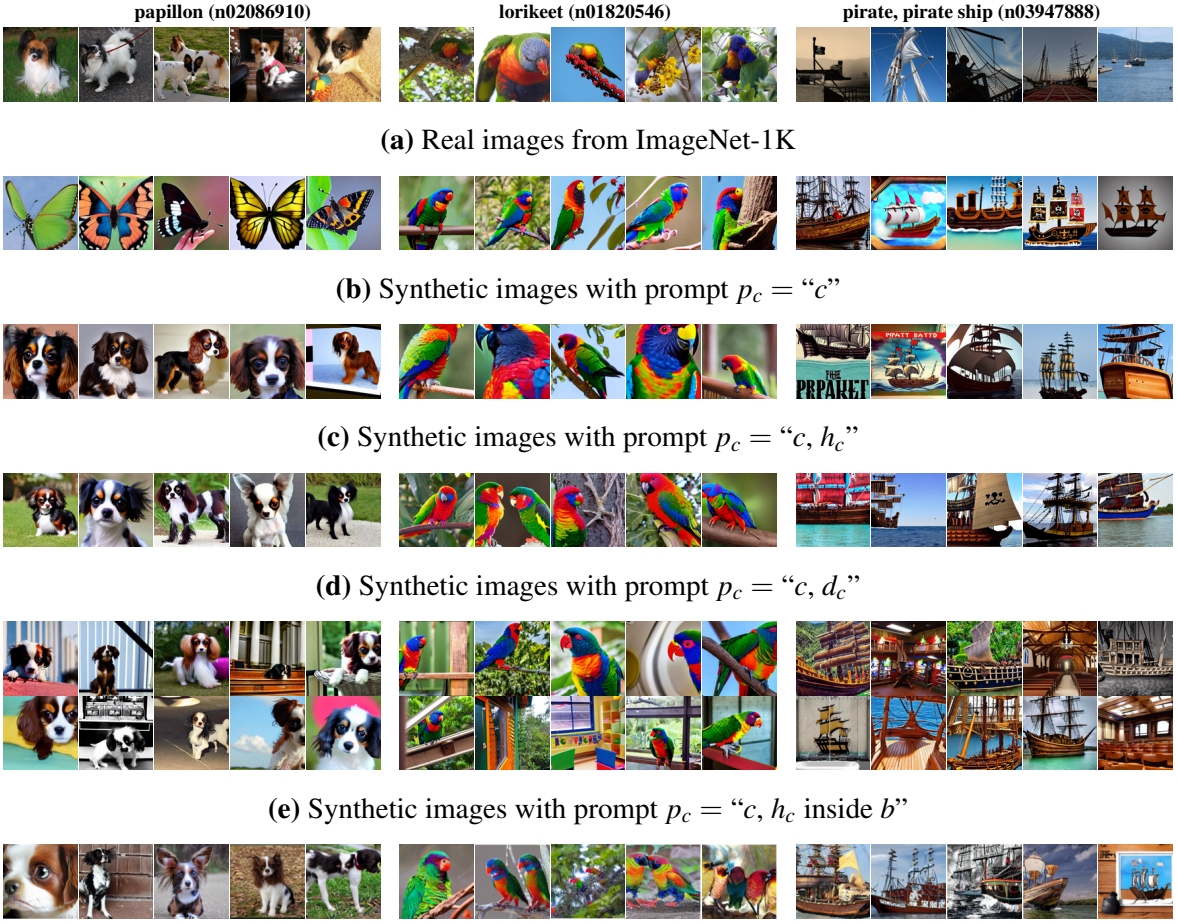
5.4 Generating synthetic ImageNet clones

For our study, we create clones of the ImageNet [Deng et al. 2009] dataset by synthesizing images depicting the classes it contains. We refer to all synthetic datasets of ImageNet classes that are created using Stable Diffusion as **ImageNet-SD**. Sec. 5.4.1 describes different ways of creating ImageNet-SD datasets starting from simply using the class name as the prompt. We then present generic, class-agnostic ways for tackling issues that arise with respect to semantics and diversity in Secs. 5.4.2 and 5.4.3, respectively. We present a few sample qualitative results in Fig. 5.3, with a more extensive set in Appendix E.

5.4.1 Generating datasets using class names

In the absence of a training set of real images, we use the generator $\mathcal{G}$ presented in the previous section to synthesize images for each class in the set $\mathcal{C}$. To do so, we need to provide the generator with at least one prompt per class. When used as an input, this class-conditioned prompt p_c triggers the generation of a synthetic image $I_c = g(p_c)$ from class c. The simplest prompt one could think of is the class name *i.e.*, $p_c =$ "c". Although CLIP [Radford et al. 2021b] uses $p_c =$ "a photo of a c" for their zero-shot experiments, using only the class name gives better results in our case.

Each class in ImageNet is associated with one or more *synsets*, *i.e.*, entities, in the WordNet [Miller 1995] graph. We use the synset lemmas corresponding to each class as class-name prompt "c", comma-separated if more than one. Fig. 5.3b shows random examples of images generated with such prompts. At first glance, one can appreciate the ability of the



(a) Real images from ImageNet-1K

(b) Synthetic images with prompt $p_c =$ "c"

(c) Synthetic images with prompt $p_c =$ "c, h_c"

(d) Synthetic images with prompt $p_c =$ "c, d_c"

(e) Synthetic images with prompt $p_c =$ "c, h_c inside b"

(f) Synthetic images with prompt $p_c =$ "c, d_c" and guidance scale parameter equal to 2.0

Figure 5.3: Qualitative results. (A) Real ImageNet images. (B)-(G) Synthetic ImageNet-SD images generated with different prompts. Despite high photo-realistic quality, some issues are noticeable for (B) such as semantic errors *e.g.*, for the class "papillon", lack of diversity, and distribution shifts *e.g.*, towards cartoons for the "pirate" class. Such issues are addressed with more expressive prompts in (C)-(D).

generator to create photo-realistic images given only a class name. In Sec. 5.5, we show that one can already obtain surprisingly good image classification results by simply training a model with this synthetic dataset.

Upon close inspection of the generated images, however, some issues become apparent: **a) semantic errors**: images generated for some classes may capture the wrong semantics (*e.g.*, see the "papillon" class in Fig. 5.3b), **b) lack of diversity**: generated images tend to look alike (an issue more apparent in Appendix E, and **c) visual domain issues**: some classes tend to shift away from natural images towards sketches or art (*e.g.*, the "pirate ship" class in Fig. 5.3b). We discuss and address these issues in the following.

5.4.2 Addressing issues with semantics and domain

As mentioned earlier, by comparing the (real) images from ImageNet with the synthetic ones generated using only synset names as prompts, we observe that for some classes their semantics do not match. This is due to polysemy, *i.e.*, multiple semantic meanings or physical instantiations of the class names we used as prompt. We show one such case in the left-most column of Fig. 5.3b: the "papillon" images correspond to butterfly for our generated dataset, while the ImageNet synset contains images of the dog breed of the same name (see Fig. 5.3a).

To reduce this semantic ambiguity, we leverage once again the fact that class names correspond to WordNet [Miller 1995] synsets. We augment the prompt for class name c with two additional elements provided by WordNet: a) The *hypernyms* h_c of the synset as defined by the WordNet graph, *i.e.*, the class name(s) of the parent node(s) of this class in the graph; and b) the *definition* d_c of the synset, *i.e.*, a sentence-length description of the semantics of each synset. In both cases, we append this information to the prompt, which becomes $p_c =$ "c, h_c" and $p_c =$ "c, d_c" for hypernyms and definition, respectively.

Qualitatively, we observed that issues regarding the semantics of the most problematic classes are fixed, and so are, to some extent, issues related to visual domain mismatch. These are also visible in Figs. 5.3c and 5.3d: appending the hypernym ($h_c =$ "toy spaniel") or the description ($d_c =$ "small slender toy spaniel with erect ears and a black-spotted brown to white coat") of the class "papillon" in the prompt produces images with the dog breed as the main subject. Appending the hypernym ($h_c =$ "ship") or the description ($d_c =$ "a ship that is manned by pirates") of the class "pirate ship" results in more natural-looking images rather than illustrations, reducing the domain shift.

5.4.3 Increasing the diversity of generated images

Generating images using more expressive prompts, *e.g.*, by appending class hypernym or definition, not only reduces semantic errors, but also increases the visual diversity of the output images. This is visible, for example, in the "lorikeet" and "pirate ship" classes in Figs. 5.3c and 5.3d when compared to Fig. 5.3b: the pose and viewpoints are slighly more diverse. However, images still tend to display the class instance centered and in a prominent position. The real ImageNet images feature significantly more diversity, several different settings and backgrounds, and, in several cases, multiple instances of the same class (*e.g.*, see Fig. 5.3a).

Although class-specific prompt engineering is an appealing option, in this study we chose to remain generic, and to increase diversity in class-agnostic ways.

Diversifying the background. We assume that class c can be seen "inside" a scene or background. To remain class-agnostic, we use all the scene classes from the Places dataset [Zhou et al. 2017] as background for every class. We generate images for every possible combination of a class c and a scene $b \in \mathcal{B}$ from the set $\mathcal{B}$ of 365 scenes in Places. We found that "c inside b" generally produces the best-looking results among a few prepositions we tried. However, we found that semantic and domain errors that arise from generating only using class name remained after specifying a background. We therefore build on top of the second simplest, but more semantically correct prompt variant, and use $p_c =$ "c, h_c inside b" to generate images in diverse scenes and backgrounds (see examples in Fig. 5.3e). Although we do not consider this in our study, selecting backgrounds tailored for each class, *e.g.*, by matching class names to scenes using features from a text encoder, seems like a promising future direction.

Reducing reliance on the textual prompt. The text-conditioned generation process of Stable Diffusion uses classifier-free diffusion guidance [Ho and Salimans 2021] which jointly trains both the conditional and unconditional diffusion models, and combines their estimates, resulting in a trade-off between sample quality and diversity. This trade-off is controlled by the guidance scale parameter, that has in practice been shown to produce high-quality images in the range of 6-9 (the default value is 7.5). Although visually detailed (see Figs. 5.3b to 5.3d), the resulting images lack diversity. We therefore experiment with reducing the guidance scale. Despite a small degradation in the visual quality of the generated images, setting the scale to 2 results in more diverse sets of images as shown in Fig. 5.3f.

Label noise and visual realism. Quite a few generated images, especially those with low guidance scale parameters or with random backgrounds (*e.g.*, see Figs. 5.3e and 5.3f) are not realistic, for example, the right-most image in the first column of Fig. 5.3f. Also, when the prompt mentions a background, some images miss the foreground object completely (*e.g.*, see the bottom row in the middle column of Fig. 5.3e) or contain impossible combinations of objects and scenes. Yet, we see such noisy or unrealistic synthetic images as a way of adding stochasticity during the training process, similar to what strong non-realistic data augmentation achieves [Geiping et al. 2023, Xu et al. 2021]. In fact, it was recently shown [Geiping et al. 2023] that diverse data augmentations, even when inconsistent with the data distribution, can be valuable (even more than additional training data) for out-of-distribution scenarios. Our experimental validation corroborates this claim.

5.5 Experiments

In this section we analyze the performance of image classification models learned using the different synthetic datasets constructed as described in Sec. 5.4. Due to the size of ImageNet-1K (roughly 1.3 million images), we perform most of our study on the smaller ImageNet-100 [Tian et al. 2020a] dataset. This allows us to run multiple flavours of each synthetic dataset and to measure the impact of several design choices. Because ImageNet-100 is a randomly chosen subset of ImageNet-1K, spanning over 100 classes and 126,689 images, it preserves some important characteristics of ImageNet-1K such as its fine-grained nature.

We denote synthetic datasets for the two ImageNet subsets as **ImageNet-100-SD** (IN-100-SD) and **ImageNet-1K-SD** (IN-1K-SD), respectively.

Experimental protocol. We follow the protocol illustrated in Fig. 5.1. The generator $\mathcal{G}$ is the Stable Diffusion [Rombach et al. 2022] v1.4 model,[2] trained on the LAION2B-en dataset [Schuhmann et al. 2022] and fine-tuned on a smaller subset filtered by an aesthetics classifier. During training, the generator is used to synthesize images for each class, which are then used for training the parameters of the encoder and the classifier. Unless otherwise stated, we create datasets of the exact same size as their real-image counterparts, *i.e.*, we generate the exact same number of images for every class as in the corresponding real dataset, maintaining any class imbalance.

We evaluate all the models on real images. When evaluating their performance over the ImageNet classes, we use both the encoder and the classifier learned during training to predict labels of real images for the 5 ImageNet datasets (Secs. 5.5.1 and 5.5.2). For transfer learning (Sec. 5.5.3), we use the pretrained encoder as a feature extractor, and learn a separate linear classifier on each of the 15 transfer datasets.

Implementation details. In all experiments, the encoder f_θ is a ResNet50 [He et al. 2016], trained for 100 epochs (unless otherwise stated) with mixed precision in PyTorch [Paszke et al. 2019] using 4 GPUs where batch norm layers are synchronized. We use an SGD optimizer with 0.9 momentum, a batch size of 256 and a learning rate linearly increased during the first 10% of the iterations and then decayed with a cosine schedule. Unless otherwise stated, we use the data augmentation pipeline from DINO [Caron et al. 2021] with 1 global and 8 local crops ($M_g = 1$ and $M_l = 8$) (see the next paragraph for an ablation on data augmentation). For Stable Diffusion we use 50 diffusion steps and a guidance scale factor of 7.5 (mostly in our ablations) or 2 (for training our models with best performance). We generate RGB images of size 512×384.

[2] https://huggingface.co/CompVis/stable-diffusion-v1-4

Training Dataset	PyTorch	DINO (+ Multi-crop)
ImageNet-100 (real)	*86.6*	*87.4* (↑ 0.80)
ImageNet-100-SD (synthetic)	28.4	43.1 (↑ **14.6**)

Table 5.1: Impact of data-augmentation for models trained on real and synthetic datasets. Performance is measured on the validation set of ImageNet-100, *i.e.*, on real images.

Impact of data augmentation. We conducted some basic experiments to evaluate the impact of different data augmentation strategies when learning from synthetic datasets. In Tab. 5.1, we report the performance of models trained on the simplest variant of ImageNet-100-SD, *i.e.*, using the class name as the prompt, utilizing either PyTorch [Marcel and Rodriguez 2010, Paszke et al. 2019] or DINO [Caron et al. 2021] augmentations. Although the gains for the real images are relatively small (less than one percent), the gains for ImageNet-100-SD are over 14%. We believe this shows two things: **a)** Synthetic images can benefit from the same augmentations as real images, and **b)** these transformations are good for domain generalization. Indeed, strong transformations have been shown to improve domain generalization [Volpi et al. 2021], and consequently can reduce the sim-to-real gap.

5.5.1 Results on ImageNet datasets

Evaluating different prompts on ImageNet-100. Tab. 5.2 compares the performance of models trained using variants of ImageNet-100-SD created with the different prompts presented in Sec. 5.4, for two different guidance scale values: 7.5 and 2. From the results for ImageNet-val and ImageNet-v2 (four left-most columns), we make the following observations: **a)** Simply using the class name as a prompt and the default guidance scale (row 2), one can synthesize images and learn a visual encoder *from scratch* that already achieves *more than* 70% *Top-5 accuracy* (43% Top-1 accuracy) on ImageNet-100, a challenging 100-way classification task with many fine-grained classes. **b)** Adding the hypernym or the definition from WordNet as part of the prompt (rows 3, 4) addresses some of the semantic and domain issues and translates into performance gains. **c)** Generating objects on diverse backgrounds (row 5), even in a simple and class-agnostic way, gives the best results for the default guidance scale, reaching over 50% Top-1 and 76% Top-5 accuracy on ImageNet-100. **d)** Using a lower guidance scale value (2) leads to more diverse image sets (as discussed in Sec. 5.4.3) and translates into the best overall performance on ImageNet-100. **e)** The exact formulation of the prompt has less impact when lowering the guidance scale; all the four prompt variants lead to similar performance as we see from rows 6-9.

Scaling the number of synthetic images. Unlike real datasets that are capped in the number of images they contain, ImageNet-SD has theoretically no size upper bound as one can

Training Dataset	R. Size	Scale		Prompt (p_c) / *Model*	IN-Val Top-1	IN-Val Top-5	IN-v2 Top-1	IN-v2 Top-5	IN-Sketch Top-1	IN-Sketch Top-5	IN-R* Top-1	IN-R* Top-5	IN-A* Top-1	IN-A* Top-5
ImageNet-100		–	1	*Baseline*	*87.4*	*96.8*	*82.5*	*95.1*	*39.1*	*58.9*	*58.4*	*79.1*	*25.6*	*68.7*
ImageNet-100-SD		7.5	2	$p_c =$ "c"	43.1	70.7	45.4	70.7	29.9	53.5	51.7	75.3	8.8	38.4
			3	$p_c =$ "c, h_c"	46.9	73.4	47.3	73.7	25.9	50.4	46.3	75.3	11.5	42.2
			4	$p_c =$ "c, d_c"	47.9	74.2	49.1	74.9	24.7	49.2	41.2	71.5	12.2	38.5
			5	$p_c =$ "c, h_c inside b"	51.5	76.8	51.2	77.4	27.9	52.5	54.0	81.8	14.1	48.4
		2.0	6	$p_c =$ "c"	63.5	86.9	62.7	86.7	41.8	**67.6**	**64.2**	83.9	13.7	45.1
			7	$p_c =$ "c, h_c"	63.4	87.1	63.5	86.5	39.2	66.7	61.9	85.1	14.9	49.1
			8	$p_c =$ "c, d_c"	64.8	86.9	65.0	87.3	33.8	60.5	51.4	77.5	14.0	48.8
			9	$p_c =$ "c, h_c inside b"	63.1	85.7	62.0	85.0	38.7	65.5	64.0	**87.2**	**21.9**	**63.1**
	10×	2.0	10	$p_c =$ "c, d_c"	72.4	90.8	70.2	90.2	40.0	65.7	55.2	79.0	15.6	53.8
	20×		11	$p_c =$ "c, d_c"	72.4	91.4	71.4	90.7	38.4	63.9	56.9	81.5	17.8	55.0
	50×		12	$p_c =$ "c, d_c"	**73.3**	**91.7**	**72.3**	**91.2**	**42.0**	67.0	59.4	82.3	17.1	57.1
ImageNet-1K		–	13	*PyTorch*	*76.1*	*92.9*	*71.1*	*90.4*	*24.1*	*41.3*	*36.2*	*52.8*	*0.0*	*14.4*
		–	14	*RSB-A1*	*80.1*	*94.5*	*75.6*	*92.0*	*29.2*	*46.5*	*40.6*	*55.1*	*11.1*	*38.6*
ImageNet-1K-SD		7.5	15	$p_c =$ "c, d_c"	26.2	51.7	26.0	51.4	9.5	22.1	15.9	32.0	2.2	10.1
		7.5	16	$p_c =$ "c, h_c inside b"	30.1	55.6	29.8	55.3	11.9	27.1	23.5	43.1	3.4	13.2
		2.0	17	$p_c =$ "c, d_c"	**42.9**	**70.3**	**43.0**	**70.3**	**16.6**	**35.1**	**26.3**	**45.3**	**3.6**	**15.1**

Table 5.2: Results on ImageNet datasets. Top-1 and Top-5 accuracy on several ImageNet datasets, namely IN-Val (the ILSVRC-2012 validation set [Russakovsky et al. 2015]), IN-v2 [Recht et al. 2019], IN-Sketch [Wang et al. 2019], IN-R [Hendrycks et al. 2021a] and IN-A [Hendrycks et al. 2021b]. In all cases, testing is done on real images. For the prompts, h_c (d_c) refers to the hypernym (definition) of class c provided by WordNet [Miller 1995], while b to scene classes from Places 365 [Zhou et al. 2017]. *IN-R and IN-A only cover a subset of the ImageNet-100 classes and we compute the reported metrics only on the common classes. Brick-colored scores denote performance higher than the models trained on real images. *Italics* denote results from models trained using real images.

generate images on demand. We therefore generated datasets which are 10×, 20× and 50× larger than ImageNet-100, using prompt $p_c =$ "c, d_c" (the best variant in Tab. 5.2, row 8) for the classes of ImageNet-100. From the last three rows of the top section in Tab. 5.2, we see that this brings gains of up to 8.5% in Top-1 accuracy on ImageNet-100, with our best model reaching 73.3% Top-1 (and 91.7% Top-5) accuracy. The gains are even more prominent for transfer learning, as we discuss in Sec. 5.5.3.

Results on ImageNet-1K. In the bottom part of Tab. 5.2 we report results on the very challenging 1000-way classification task of ImageNet-1K (IN-Val) that contains many fine-grained categories of mushrooms, birds and dogs [Huh et al. 2016]. We see that the model trained on our synthetic ImageNet-1K-SD dataset using the prompt composed of the class name and description ($p_c =$ "c, d_c") and using guidance scale 2 reaches 42.9% Top-1 and 70.3% Top-5 accuracy on the ImageNet-1K validation set. Although significantly lower than the results achieved by a model trained on the 1.3 million real images of ImageNet, we see

Training Dataset	Scale		Prompt (p_c) / *Model*	Aircraft	Cars196	DTD	EuroSAT	Flowers	Pets	Food101	SUN397	iNat18	iNat19	Avg.
–	–	1	Random Weights	11.9	3.7	17.0	73.1	26.9	11.9	13.3	7.3	0.1	1.3	16.6
ImageNet-100	–	2	*Baseline*	*43.6*	*41.5*	*67.9*	*96.2*	*85.6*	*78.7*	*63.4*	*51.2*	*22.8*	*33.4*	*58.4*
ImageNet-100-SD	2.0	3	$p_c =$ "c, d_c" (50×)	**47.9**	**44.5**	**74.0**	**96.8**	**89.6**	**83.7**	**68.6**	**57.2**	**29.5**	**40.6**	**63.2**
ImageNet-1K	–	4	*PyTorch*	*48.9*	*49.9*	*72.1*	*96.2*	*89.3*	*92.3*	*71.2*	*60.5*	**35.5**	*41.5*	*65.7*
	–	5	*RSB-A1*	*46.8*	*54.4*	*73.8*	*95.8*	*88.6*	**93.0**	*71.3*	**63.4**	*34.9*	*43.2*	*66.5*
ImageNet-1K-SD	7.5	6	$p_c =$ "c, d_c"	48.7	49.7	71.6	96.5	90.1	81.9	66.4	55.8	28.7	40.6	63.0
	7.5	7	$p_c =$ "c, h_c inside b"	49.6	47.4	72.1	95.9	89.3	87.2	67.7	59.5	30.8	41.4	64.1
	2.0	8	$p_c =$ "c, d_c"	**55.3**	**57.2**	**75.9**	**96.7**	**92.9**	88.7	**73.1**	62.5	35.0	**46.3**	**68.4**

Table 5.3: Top-1 accuracy on ten transfer learning datasets for encoders trained on real and synthetic images. We treat encoders as feature extractors and train linear classifiers on top for each dataset. Brick-colored scores denote performance higher than the models trained on real images. We make the remarkable observation that representations from models trained on synthetic data can match the generalization performance of representations from models trained on millions of real images. *Italics* denote results from models trained using real images.

that the synthetic dataset is able to at least partially capture the subtle clues needed to differentiate fine-grained classes. Similar observations can be made on ImageNet-v2 [Recht et al. 2019] (IN-v2).

5.5.2 Resilience to domain shifts

We investigate the performance of our models on three challenging evaluation sets for ImageNet-1K classes: ImageNet-Sketch [Wang et al. 2019] (IN-Sketch), ImageNet-R [Hendrycks et al. 2021a] (IN-R) and ImageNet-A [Hendrycks et al. 2021b] (IN-A). These datasets contain out-of-distribution images and their goal is to test resilience to domain shifts and adversarial images. Results are reported in the right-most columns of Tab. 5.2.

For ImageNet-100, we see from the top part of the table that a number of ImageNet-100-SD models *outperform* the model trained on real images for ImageNet-Sketch and ImageNet-R. The best Imagenet-100-SD model, *i.e.* the one trained with 50× images, further rivals the baseline on ImageNet-A.

When it comes to a much harder classification task like the 1000 classes of ImageNet-1K, we see from the lower part of Tab. 5.2 that the same trend does not really hold. The ImageNet-1K-SD model trained on synthetic data lags behind in all cases when compared to the two models [Paszke et al. 2019, Wightman et al. 2021] that are trained on the ImageNet-1K training set.

5.5.3 Transfer learning

In previous evaluations, we used pretrained models as a whole, *i.e.*, encoders together with classifiers, all trained on synthetic ImageNet datasets, and we directly applied those to predict the label of the (real) test images on the training classes. Here, we use a slightly different protocol. We evaluate the quality of the representations learned by our encoders alone, by using them as feature extractors and training linear logistic regression classifiers from scratch on top as done in transfer learning [Kornblith et al. 2019, Sariyildiz et al. 2021].

We report results on 15 transfer datasets: **a)** eight common small-scale datasets (Aircraft [Maji et al. 2013], Cars196 [Krause et al. 2013], DTD [Cimpoi et al. 2014], EuroSAT [Helber et al. 2019], Flowers [Nilsback and Zisserman 2008], Pets [Parkhi et al. 2012], Food101 [Bossard et al. 2014], SUN397 [Xiao et al. 2010]), **b)** two long-tail datasets (iNat2018 [Van Horn et al. 2018] and iNat2019 [Van Horn et al. 2018]), and **c)** the five datasets ("levels") of the CoG benchmark [Sariyildiz et al. 2021]. For each of the transfer datasets, we first extract features from the pretrained encoders and then train linear logistic regression classifiers using these features. For the larger transfer datasets, *i.e.*, iNaturalist 2018 [Van Horn et al. 2018] and iNaturalist 2019 [Van Horn et al. 2018] datasets and the CoG levels, we train linear classifiers in PyTorch [Paszke et al. 2019] using SGD, following [Sariyildiz et al. 2021]. For the remaining 8 smaller transfer datasets, we follow [Kornblith et al. 2019] and train classifiers using L-BFGS implemented in Scikit-learn [Pedregosa et al. 2011]. In all cases, we resize the images with bicubic interpolation so that their shortest side is 224 pixels, and then take a central crop of 224×224 pixels. We tune hyper-parameters (learning rate and weight decay for the SGD optimizer, and regularization coefficient for the L-BFGS optimizer) using Optuna [Akiba et al. 2019] over at least 25 trials. Code for evaluations can be found here[3].

We report Top-1 accuracy on the (real) test set of the small-scale and long-tail datasets (10 datasets in total) in Tab. 5.3. We compare ImageNet-100-SD and ImageNet-1K-SD visual encoders obtained with some of our best prompts to baselines trained on ImageNet-100 and ImageNet-1K. What we observe is quite striking: On average, representations learned on purely synthetic images exhibit *generalization performance comparable to representations trained on thousands or millions of real images*. This suggests that synthetic images can be used to pretrain strong general-purpose visual encoders.

We also evaluate our best ImageNet-SD model on the ImageNet-CoG benchmark introduced in [Sariyildiz et al. 2021] to measure concept generalization and report Top-1 accuracy obtained on the test sets of these datasets in Tab. 5.4. We compare the performance of the best ImageNet-1K-SD model (from Tab. 5.3) to strong baselines trained on ImageNet-1K like the supervised RSB-A1 [Wightman et al. 2021] and self-supervised DINO [Caron et al. 2021]

[3] https://github.com/naver/trex/tree/master/transfer

Training Dataset	Prompt (p_c) / Model	IN-1K	L_1	L_2	L_3	L_4	L_5
ImageNet-1K	*PyTorch*	*75.8*	*67.8*	*63.1*	*58.9*	*58.2*	*52.0*
	RSB-A1	***79.8***	*69.9*	*65.0*	*60.9*	*59.3*	*52.8*
	DINO (self-supervised)	*74.8*	***71.1***	***67.2***	***63.2***	***62.6***	***57.6***
ImageNet-1K-SD	$p_c =$ "c, d_c"	70.4	65.7	61.8	58.5	58.0	52.4

Table 5.4: Top-1 accuracy on the ImageNet-CoG benchmark [Sariyildiz et al. 2021]. We report performance for the best ImageNet-1K-SD model from Tab. 5.3 (with guidance scale equal to 2), and compare it to the state-of-the-art supervised and self-supervised models trained on the real images of ImageNet-1K, RSB-A1 [Wightman et al. 2021] and DINO [Caron et al. 2021], respectively.

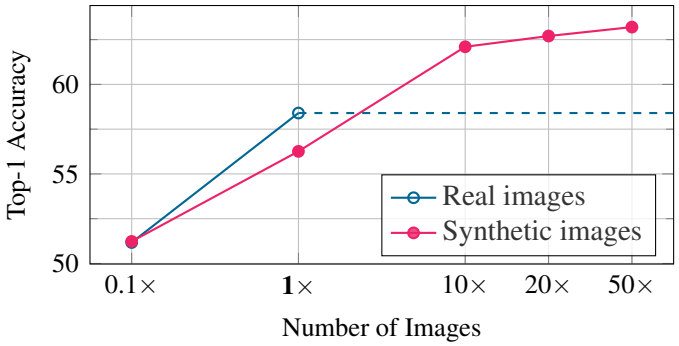
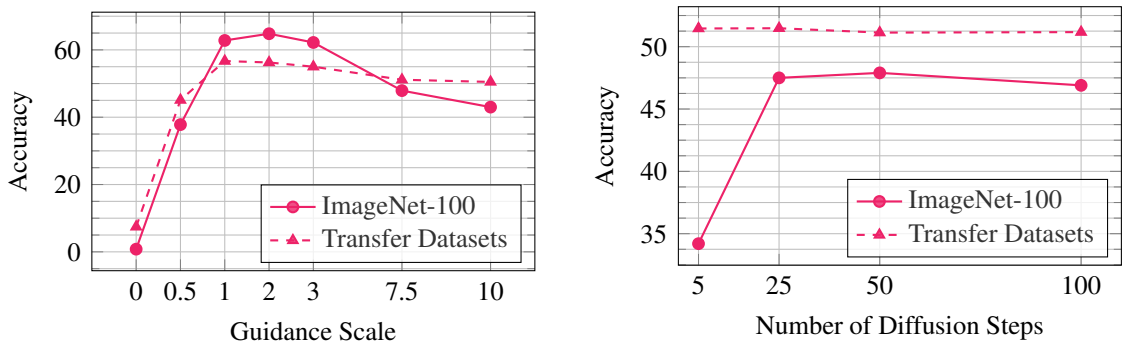


Figure 5.4: Scaling the number of training images. Average Top-1 accuracy on 10 transfer datasets (from Tab. 5.3) when training on ImageNet-100 using $(1/10)$-th to 50× images (relative to the real dataset size).

models. We observe that on L_5, which is the most challenging level, the performance of the representations learned on synthetic images is comparable to that of learned on real images by supervised models (*i.e.*, PyTorch and RSB-A1). As we move towards L_1, we see that the gap between these two models increases in favor of RSB-A1. Finally, after training classifiers (only) using the real images of IN-1K, our model reaches 70.4% accuracy, significantly closing the gap to even the most optimized models trained on real data like RSB-A1. This protocol differs from the one presented in Sec. 5.5.1 as it uses real images to train a linear classifier on top of the feature extractor trained only on synthetic images, hence the IN-1K results are not comparable with Tab. 5.2.

Scaling the number of synthetic images for transfer. Fig. 5.4 reports transfer learning performance on the 10 datasets of Tab. 5.3, when varying the size of the training set. We see that generating 10× more images allows the ImageNet-100-SD model to outperform the model trained on real images, and the gains increase as we generate up to 50× more.



(a) Impact of the guidance scale parameter

(b) Impact of the nubmer of diffusion steps

Figure 5.5: Impact of the guidance scale parameter and number of diffusion steps. Top-1 accuracy on ImageNet-100 and averaged over 10 transfer datasets (from Tab. 5.3) for $p_c =$ "c, d_c". In the left plot, steps are set to 50, in the right plot guidance scale is 7.5.

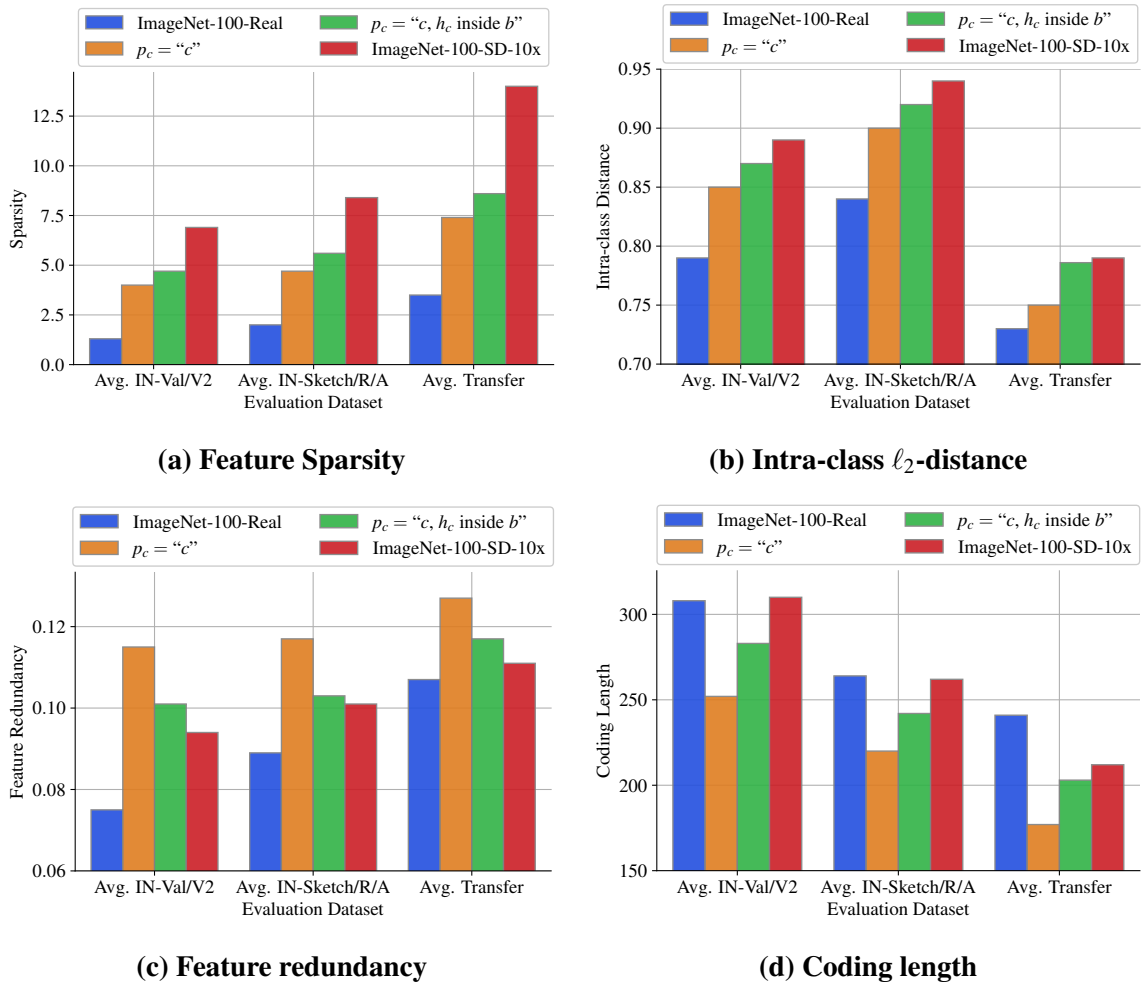
5.5.4 Impact of guidance scale and diffusion steps

In Fig. 5.5 we analyse the impact of the guidance scale and diffusion step hyper-parameters of Stable Diffusion [Rombach et al. 2022]. As we discussed earlier, a lower guidance scale leads to more visual diversity and that is reflected of performance. Values of 1 to 3 all seem like a good choice. When it comes to the number of diffusion steps, values like 25 and (the default) 50 seem like a safe choice, with 25 being slightly worse, but requiring half the time to extract. Interestingly, using more steps seems to slightly hurt performance on the training classes. It is worth noting that transfer learning performance is surprisingly and consistently high for even 5 diffusion steps. This corroborates recent finding that training on complex but possibly semantically meaningless images like fractals [Kataoka et al. 2022] or sinusoidal waves [Takashima et al. 2023] can provide a strong starting point for visual representations that generalize well.

5.5.5 Analysis of the learned features

In this section, we analyze and contrast the *representations* obtained with models we trained using synthetic images to representations from models trained on real images. For this analysis, we used ImageNet-SD models for images that were generated using the default prompt guidance scale of Stable Diffusion, *i.e.*, 7.5. We perform our analysis for ImageNet-100 and using four metrics: **a)** sparsity, **b)** intra-class distance, **c)** feature redundancy and **d)** coding length.

We compare four different models trained on either real or synthetic data for the 100 classes of ImageNet-100: One model trained on real images, ImageNet-100-Real, two models trained on synthetic image sets of the same size obtained by using two different prompts: $p_c =$ "c"



(a) Feature Sparsity

(b) Intra-class ℓ_2-distance

(c) Feature redundancy

(d) Coding length

Figure 5.6: Feature analyses for models. We perform these analyses on top of features extracted from pretrained encoders f trained on either real or synthetic data for ImageNet-100 (training data is specified in the legends of the subfigures). For the purpose of this study, we use synthetic data generated with guidance scale equal to 7.5. *Sparsity* is measured by the percentage of dimensions close to zero [Kornblith et al. 2021]. *Intra-class ℓ_2-distance* is the average pairwise ℓ_2-distance between samples from the same class. These two metrics are computed on ℓ_2-normalized features. *Feature redundancy* [Wang et al. 2022b] is obtained by $\mathcal{R} = \frac{1}{d^2}\sum_i \sum_j |\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})|$, where $\mathbf{X} \in \mathbb{R}^{N \times d}$ is a feature matrix containing N samples, each encoded into a d-dimensional representation (2048 in our case) and $\rho(\mathbf{X}_{:,i}, \mathbf{X}_{:,j})$ is the Pearson correlation between a pair of feature dimensions i and j. *Coding length* [Yu et al. 2020] is measured by $R(\mathbf{X}, \varepsilon) = \frac{1}{2}\log\det(\mathbf{I}_d + \frac{d}{N\varepsilon^2}\mathbf{X}^\top\mathbf{X})$, where $\mathbf{I}_d$ is a d-by-d identity matrix, ε^2 is the precision parameter set to 0.5.

and $p_c =$ "c, h_c inside b", and the ImageNet-100-SD-10x model, trained using ten times more images.

We perform these analyses on all the datasets considered in this work, except for the 5

ImageNet-CoG levels. For the sake of this study, we split them into three groups: **a)** ImageNet-100-Val/v2, **b)** ImageNet-100-Sketch/A/R and **c)** the 10 transfer datasets (long-tail and small-scale). For each pretrained model and dataset, we extract features for either only the images in the test set (for the ImageNet test sets), or for all images (for the small transfer datasets). We then compute each of the four metrics separately on each dataset, and average them over all datasets in the same group. Before computing metrics, we ℓ_2-normalize features.

Results of the analyses for each of the four metrics are as follows.

Sparsity. Inspired by Kornblith et al. [2021], we compute feature *sparsity ratio*, *i.e.*, the percentage of feature dimensions close to zero with a threshold of 10^{-5}. We report sparsity ratios in Fig. 5.6a. We see that the sparsity ratio for the models trained on synthetic images increases as the "diversity" of a synthetic dataset increases, *i.e.*, we see gradual increase in sparsity scores from $p_c =$ "c" and $p_c =$ "c, h_c inside b" to ImageNet-100-SD-10x. This observation aligns with their performance as well, *i.e.*, in Tab. 5.2 we show that ImageNet-100-SD-10x performs best in general (among the 3 variants considered in this analysis) while $p_c =$ "c" performs worst. More interestingly, we see that ImageNet-100-Real, the model trained on real images, learns the most sparse representations.

Intra-class distance. In Sec. 5.4, we present simple ways to increase the diversity of synthetic images. Now we check if these efforts increase the variance of samples in the representation space. To do that, we compute the average ℓ_2-distance between samples from the same class (*i.e.*, intra-class distance). We see in Fig. 5.6b that models trained with more diverse images indeed learn representations with higher intra-class variance.

Feature redundancy. Following Wang et al. [2022b], we compute feature redundancy, *i.e.*, average pairwise Pearson correlation among dimensions. From Fig. 5.6c we see that the redundancy of features learned on real images increase more rapidly than the ones learned on synthetic images, as we move from ImageNet-100-Val/v2 towards out-of-domain or transfer datasets.

Coding length. To further investigate our observation on feature redundancy, we follow Yu et al. [2020] and compute the average coding length per sample on each dataset (see Fig. 5.6d). We see that models trained on ImageNet-100-Real and ImageNet-100-SD-10x are comparable.

5.6 Discussion

This section takes a step back and considers some of the implications from the analysis proposed in this chapter.

Applicability beyond ImageNet. The process we followed to create ImageNet-SD requires minimal assumptions and can be applied to a wider set of classes. To disambiguate semantics, we only assume access to a short textual description of the class. This is generally easy to acquire even at a larger scale, *e.g.*, in semi-automatic ways from Wikipedia.

Scaling laws for synthetic data. Conceptually, there is no reason to restrict our approach to a finite dataset of synthetic images. We could devise a training process which sees each image only once [Parisi et al. 2019].

Yet, despite this scaling potential, the quality of the resulting classifier is bounded by the expressivity of the generator and the concepts it can reliably reproduce. No matter how intriguing the promise of an "infinite dataset" via data generation might be, practical applications are bound by costs linked to computation and storage, as well as the moderation of the content fueling this generator. The latter has strong implications we discuss next.

Data and model bias. Because of its pioneering role as a source of images to train generic models, and all it has done to advance the computer vision field, ImageNet and some of its bias has been under heavy scrutiny [Denton et al. 2021, Luccioni and Rolnick 2022]. Its synthetic counterparts have no reason to be immune to bias.

The main advantage of training with synthetic dataset is also its biggest flaw. Instead of manually curating and annotating a dataset, this process is outsourced to a text-to-image generator, whose training data is not always known. Our study is based on the text-to-image generator of Stable Diffusion (SD). SD is trained on LAION-2B [Schuhmann et al. 2022], a dataset scraped from the internet and filtered in an automatic way using CLIP [Radford et al. 2021b]. LAION has been shown to contain problematic content [Birhane et al. 2021] and SD models to memorize at least part of the training set [Carlini et al. 2023, Somepalli et al. 2022]. Algorithmic bias is not only due to bias in the data [Hooker 2021], yet biased datasets lead to biased models and predictions [Aka et al. 2021, Salman et al. 2022, Steed and Caliskan 2021]. Frameworks such as [Hutchinson et al. 2021] could be considered to increase transparency and accountability.

On top of the bias in the data, the architecture itself constraints the generated images, and as such, propagates and potentially amplifies [Bianchi et al. 2022] existing bias. A major one that we have discussed earlier is the lack of diversity. An obvious corollary is the fact that stereotypes are reinforced. The options we have explored mitigate this issue to some

limited extent, in that it improves classification results, but this issue is far from being solved. Finally, there are many societal implications of using such models to generate synthetic datasets for training computer vision models, and a more thorough and multi-disciplinary discussion is required.

5.7 Conclusions

In this chapter, we study to which extent ImageNet, arguably the most popular computer vision dataset, can be replaced by a dataset synthesized by a text-to-image generator. Through an extensive study, we find that one can learn models that exhibit surprisingly good performance on fine-grained classification tasks like ImageNet-100 and ImageNet-1K without any class-specific prompting. However, the most important result of this study is the finding that models trained on synthetic data exhibit exceptional generalization capability that rivals with models learned with real images. We see this study as merely a first glimpse of what is now possible with the latest large models in terms of visual representation learning. We envision that similar approaches could be used to fine-tune or adapt models, using those synthetic datasets side-by-side with real ones.

Chapter 6

Conclusion

Contents

6.1 Summary of contributions 99
6.2 Perspectives for future work 101

As concluding remarks, we summarize our contributions (in Sec. 6.1) and present perspectives for future research (in Sec. 6.2).

6.1 Summary of contributions

In this thesis, we have studied image representation learning, and made three contributions on evaluating the transferability of representations learned on ImageNet-1K (the first) and learning more transferable representations on ImageNet-1K (the second and the third):

1. Measuring concept generalization in visual representation learning,
2. Improving the generalization of supervised learning models and
3. Learning transferable representations from synthetic ImageNet clones

We summarize these works and our main conclusions in the following.

Measuring concept generalization in visual representation learning

In Chapter 3, we investigated the problem of evaluating the concept generalization performance of representations learned on ImageNet-1K, when controlling other generalization facets (*i.e.*, domain or task generalization). We identified the need for a benchmark, and introduced ImageNet-CoG. In ImageNet-CoG, the seen concepts are from ImageNet-1K [Russakovsky et al. 2015], and there are five disjoint sets of unseen concepts (which we call "CoG levels", *i.e.*, L_1, L_2, L_3, L_4, L_5) sampled from the remainder of the full ImageNet dataset [Deng et al. 2009]. Moreover, our ImageNet-CoG benchmark has the following unique properties: **a)** both the seen and unseen concept sets are part of the full ImageNet dataset, hence, the same concept ontology, *i.e.*, WordNet [Miller 1995], **b)** semantic similarity between each seen and unseen concept is measured by Lin similarity defined on the WordNet ontology, and **c)** from the first to the last level, each level contains unseen concepts

that are semantically less and less similar to the seen ones in ImageNet-1K. As our benchmark is conveniently applicable to any model trained on ImageNet-1K, we evaluated 31 popular representation learning models, including self-supervised learning models, supervised learning models trained with different architectures or regularization techniques and weakly-supervised models, which were first pretrained on less-curated web-data. Our most interesting observations are as follows:

- It is harder for models to generalize to semantically distant concepts and our CoG levels are increasingly challenging transfer datasets, *i.e.*, the performance of all the models decreases as we evaluate them on L_1 through L_5.
- Self-supervised models excel at concept generalization. They are more resilient than supervised models to the semantic concept shift.
- Label-associated augmentation techniques deteriorate concept generalization performance, although they improve the performance on the seen concepts.
- Transformer-based models appear to overfit to seen concepts, unlike neural architecture-search-based models, which seem promising for concept generalization.

Improving the generalization of supervised learning models

In Chapter 4, we proposed a new training setup for supervised learning of visual representations on ImageNet-1K. Our setup was inspired mainly by the recent findings of Kornblith et al. [2021] and some of our observations from the ImageNet-CoG benchmark evaluation listed above, *i.e.*, stronger supervised models on ImageNet-1K learn less transferable representations to other concepts. By reinforcing supervised learning with advances in self-supervised learning, more specifically, with multi-crop augmentation [Caron et al. 2020], expendable projector head [Chen et al. 2020a] and momentum encoders [He et al. 2020], we improved the generalization performance of supervised models on more than 15 transfer datasets, including our CoG levels. Moreover, by substituting trainable class weights in our models with prototypes obtained over a memory bank of representations we boosted performance even further. Our main observations are as follows:

- The trade-off between training (ImageNet-1K classification) and transfer performance can be controlled by the size of projectors, *i.e.*, the bigger the projectors, the better the transfer performance.
- Image labels (if available) can be used to improve the utility of models for transfer tasks, *i.e.*, our best models outperform state-of-the-art self-supervised models on transfer tasks while still being significantly better than them on ImageNet-1K.

- Our simple training setup also achieves state-of-the-art performance on the ImageNet-1K classification task.
- The models leading on each end of the trade-off between training and transfer performance learn complementary representations that can be combined, *e.g.*, via feature concatenation, or prediction averaging.

Learning transferable representations from synthetic ImageNet clones

In Chapter 5, we investigated the extent to which synthetic images could help learning useful visual representations. To this end, we generated synthetic ImageNet-1K clones (which we called ImageNet-1K-SD) via Stable Diffusion [Rombach et al. 2022], a recent text-to-image generative model. Then we trained supervised models on these synthetic clones. To generate synthetic images for each of the 1000 classes in ImageNet-1K, we used simple textual prompts with class information, such as the name or description of a class. After training our models on ImageNet-1K-SD, we evaluated them on real images of 5 ImageNet datasets from different domains and 15 transfer datasets, and observed the following:

- Our models trained on synthetic ImageNet-1K-SD demonstrate overall decent performance when tested on the real ImageNet datasets, but fall behind the baseline model trained on the real images of ImageNet-1K. We still find these results promising, as they are obtained with synthetic images that are generated via a all-purpose generative model (without fine-tuning on ImageNet-1K) and without any extensive prompt engineering.
- More interestingly, representations learned by our models exhibit notable generalization capability when tested on the 15 transfer datasets, *i.e.*, their performance is comparable to (and in some cases, better than) the baseline models learned with real images.

6.2 Perspectives for future work

We conclude the thesis with a discussion on the limitations of our contributions and possible directions for future work.

Going beyond ImageNet-1K training

In all the three contributions presented in this thesis, we either trained models on ImageNet-1K (in Chapters 4 and 5) or evaluated models that are pretrained on ImageNet-1K (in Chapter 3). We primarily used ImageNet-1K, because it is a well-established dataset for large-scale representation learning. However, it is important to note that ImageNet-1K has its

own biases. It is an iconic [Zhang et al. 2014] and object-centric [Torralba and Efros 2011] dataset, where each image usually contains a canonical view of a single object of interest. Additionally, it is curated and balanced, *i.e.*, images are manually annotated and the number of images for each of the 1000 classes is similar.

While these properties make ImageNet-1K a good choice for studying representation learning, they also limit its suitability for computer vision tasks in other domains where these assumptions are not always valid. For instance, in robotics, autonomous driving or satellite imagery [Bourcier et al. 2022b] problems, images are often scene-centric [Zhou et al. 2017] and the frequency of objects present in scenes might follow a long-tail distribution. As a result, benchmarks initially designed for evaluating ImageNet-1K training or models developed for learning representations on ImageNet-1K might not be directly applicable to these scenarios. In this regard, we believe that extending our contributions to such scenarios would be an interesting direction for future work. In the following, we discuss a few starting points.

- Studying concept generalization on scene-centric datasets, such as Places [Zhou et al. 2017], would be not only interesting, but also more challenging. For instance, seen and unseen scene splits could be defined in two ways. First, by considering the objects present in the scene, *i.e.*, a scene is considered seen if it contains at least one object that is present in the training split of the dataset, and unseen otherwise. In this case, as a scene can contain multiple objects, one needs to make sure to annotate all possible objects in the scene. Second, by considering the scenes themselves, *i.e.*, treating each scene as a concept and partitioning the set of all scenes into disjoint seen and unseen scenes. This way, it could be possible to measure generalization to unseen objects or scenes, separately. As for the semantic similarity measure, in addition to computing the ℓ_2 distance between language model embeddings of objects or scenes (similar to what we presented in Chapter 3), one can also consider exploiting the scene graph information [Krishna et al. 2017] or using simpler features, like co-occurrence of objects in a scene.

- Multi-crop augmentation, which is one of the components of our improved training setup for supervised models, suits well to object-centric images, *i.e.*, it is reasonable to assume that taking small random crops from the image will contain a part of the object of interest. But this assumption does not hold for scene-centric images, *i.e.*, small random crops taken from a scene-centric image might contain different objects. On one hand, representations might be biased to individual objects instead of the scene as a whole, and the model might struggle to differentiate between scenes which contain similar sets of objects. On the other hand, due to multi-crop, the model might better learn the context of objects in a scene, improving its overall performance on

scene understanding tasks. Future research can explore the trade-offs and potential best practices for multi-crop augmentation in scene-centric images.

- While generating synthetic ImageNet clones, we prepared textual prompts (to be given as input to Stable Diffusion) tailored for only one concept of ImageNet-1K. An alternative approach could be to consider prompts mentioning multiple different concepts at the same time and generate synthetic images for a scene. This would serve as an additional data augmentation operation to increase the diversity of synthetic images, acting in a similar manner to Mix-Up [Zhang et al. 2018] or CutMix [Yun et al. 2019]. But coherently generating images for multiple concepts at the same time would be a more challenging task. First, the group of concepts that make up a scene must be selected carefully, as the model might have biases towards certain co-occurrences of concepts (hence, producing low-quality images for combinations of concepts not known to the model). Moreover, as the prompts mentioning multiple concepts would be longer and more complex, the generative model should be capable of generating images that are consistent with such prompts. Both concerns might impact the visual fidelity of synthetic images, and hence, the quality of the learned representations. Another use case for synthetic images could be to overcome the data imbalance problem by generating images for the concepts not well represented in the training set, which shares similar motivations with, for example, zero-shot learning models [Sariyildiz and Cinbis 2019]

Going beyond image classification as a training or transfer task

In our contributions, we considered exclusively the image-level classification tasks, for both training our models and evaluating their learned representations. However, other pixel-level tasks can also benefit from all-purpose visual representations, such as object detection or segmentation [Kirillov et al. 2023]. In the following, we envision a few directions to explore tasks beyond image classification, specifically object detection or segmentation which necessitate that models to not only recognize the object of interest in the image, but also localize or segment it.

- It would be interesting to investigate the concept generalization capabilities of representations for these tasks, *i.e.*, how well a model can localize or segment a concept it has never seen before. If, for instance, a model exhibits good localization or segmentation performance for unseen concepts, albeit its poor recognition accuracy, it would be a strong indication that recognition as a task could be disentangled from localization or segmentation, and this might inspire further research on developing meta-detection or segmentation models. To study this, though, one needs to gather appropriate annotations (bounding boxes for detection and segmentation masks) for images of both seen and unseen concepts (for the 1000 ImageNet-1K and 5000 ImageNet-CoG concepts

in our case), which can be a laborious task. One potential solution is to use crowdsourcing platforms, such as Amazon Mechanical Turk, to gather these annotations.

- Investigating the impact of multi-crop augmentation or expendable projectors on the generalization performance for object detection or segmentation is another interesting direction. First, training on small crops which contain only a part of the object of interest might enforce representations to capture the details of the object as a whole rather than focusing only on the most discriminative part of the object. This, in turn, would allow the model to better distinguish objects from each other and from background. Second, expendable projectors would prevent the model from overfitting to localizing or segmenting seen concepts, and hence, improve its generalization performance on unseen ones.

- Generating synthetic images for object detection or segmentation is another potentially interesting direction. Recent image generative models (including Stable Diffusion) can condition the image generation process on diverse forms of side information, such as segmentation masks or scene layouts [Rombach et al. 2022]. By leveraging this capability, one can generate synthetic object detection or segmentation datasets for, *e.g.*, the concepts in the MS-COCO dataset [Lin et al. 2014]. Then the real images of MS-COCO can be used to evaluate the generalization capability of learned representations.

In summary, this thesis explored various aspects of visual representation learning, with a focus on evaluating concept generalization, improving the generalization of supervised models, and using synthetic images for training models. While our contributions provide valuable insights, there are still several avenues for future research as discussed earlier.

Appendix A

ImageNet-CoG with the ImageNet 2021 release

The ImageNet team recently released a new version of IN-21K as well as the ILSVRC-2012 dataset (IN-1K)[1]. Both datasets are available for download directly from the official website[2].

Dataset	Split	# Images
IN-1K [Russakovsky et al. 2015]	train	1281167
IN-1K [Russakovsky et al. 2015]	val	50000
IN-1K-blurred [Yang et al. 2021]	train	1281066
IN-1K-blurred [Yang et al. 2021]	val	49997

Table A.1: Comparison of the number of images in IN-1K and IN-1K-blurred.

The 2021 Winter version of IN-21K. We built ImageNet-CoG on the 2011 Fall release of IN-21K, which was the only version available in 2020, when we started constructing our benchmark. The 2011 Fall version contained 21841 concepts, while the new release has only 19167 concepts–a subset of the concepts from the Fall 2011 release. This follows recent studies from the ImageNet team, which identify potentially problematic concepts [Yang et al. 2020]. Such concepts were removed from the latest ImageNet version, including all the concepts under the "Person" sub-tree in WordNet.

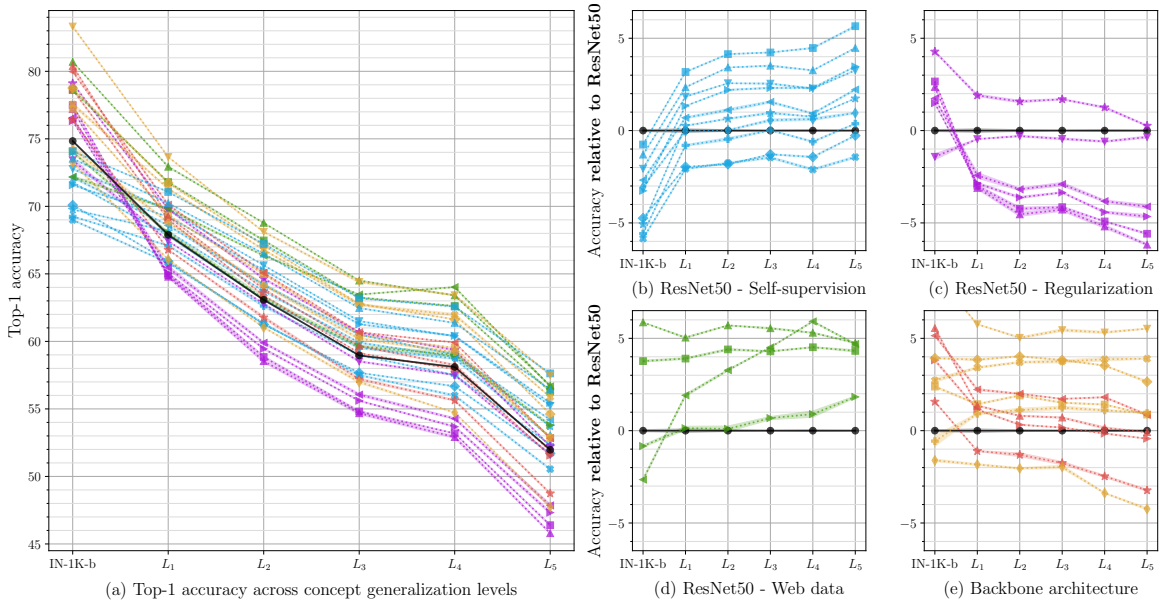
With this modified version we successfully verified that: i) all the concepts of ImageNet-CoG are available in the new release, and ii) the images for all the 5000 concepts of ImageNet-CoG are identical in both releases. Consequently, all the results in our work *can also be reproduced* using the Winter 2021 version of IN-21K.

Blurred version of IN-1K. To protect the privacy of people present in some of the IN-1K images, the ImageNet team released a new version of this dataset, which we refer to as IN-1K-blurred [Yang et al. 2021]. In this version, the faces of people are blurred in the images. The statistics of these two versions are compared in Tab. A.1 in Appendix.

[1] https://image-net.org/update-mar-11-2021.php
[2] https://image-net.org/download-images.php

ResNet	Transformer	NAS & Other
● ResNet50 (23.5M) ■ *a*-ResNet152 (58.1M)	▲ *a*-T2T-ViT-t-14 (21.1M) ▶ *a*-DeiT-S (21.7M) ◀ *a*-DeiT-S-distilled (21.7M) ▼ *a*-DeiT-B-distilled (86.1M)	★ *a*-Inception-v3 (25.1M) ✚ *a*-EfficientNet-B1 (6.5M) ✖ *a*-EfficientNet-B4 (17.5M) ◆ *a*-NAT-M4 (7.6M) ⧫ *a*-VGG19 (139.6M)

Self-Supervision		Web data	Regularization	
■ *s*-DINO ▲ *s*-SwAV ▶ *s*-BarlowTwins ◀ *s*-OBoW ▼ *s*-BYOL	★ *s*-SimCLR-v2 ✚ *s*-MoCo-v2 ✖ *s*-MoCHi ◆ *s*-CompReSS ⧫ *s*-InfoMin	■ *d*-Semi-Sup ▲ *d*-Semi-Weakly-Sup ▶ *d*-MoPro ◀ *d*-CLIP	■ *r*-ReLabel ▲ *r*-CutMix ▶ *r*-MixUp ◀ *r*-Manifold-MixUp	▼ *r*-Adv-Robust ★ *r*-MEAL-v2



(a) Top-1 accuracy across concept generalization levels

(b) ResNet50 - Self-supervision

(c) ResNet50 - Regularization

(d) ResNet50 - Web data

(e) Backbone architecture

Figure A.1: Linear classification on ImageNet-CoG using blurred images for IN-1K. Top-1 accuracies for all the 31 models listed in Tab. 3.2, after training logistic regression classifiers on the **blurred version** of IN-1K (IN-1K-b in the plots) and each level $L_{1/2/3/4/5}$. (a) Absolute Top-1 accuracy on all levels. (b)-(e) accuracy relative to the baseline ResNet50 for all the models, split across the four model categories presented in Sec. 3.4.1.

Although the models we evaluated in this work were pretrained on IN-1K, with non-blurred images, for future reference, we performed our evaluation also on the blurred version of IN-1K (IN-1K-blurred) for all the models. Concretely, for each model, we follow our evaluation protocol on IN-1K-blurred by extracting features of the blurred images and training logistic regression classifiers on them. We report these results in Fig. A.1 in Appendix. Note that Fig. A.1 is the new version of Fig. 3.4 with results obtained on IN-1K-blurred instead of IN-1K. We observe that the scores drop on average 0.91%, which is comparable to the 0.68% drop observed on popular models [Yang et al. 2021].

Appendix B

ImageNet-CoG extended results

In Sec. 3.4, we evaluate concept generalization performance for 31 models (listed in Tab. 3.2) on ImageNet-CoG. Figs. 3.4 and 3.5 report the results of training logistic regression classifiers with all the available training data for each concept (discussed in Sec. 3.4.2.1), and training it with a few samples per concept (discussed in Sec. 3.4.2.2), respectively. Although Fig. 3.4 includes the results for all the models on all concept generalization levels, Fig. 3.5 provides only a selection of the most interesting few-shot results. In this section, we present the full set of results for all the methods when training with few and all data samples in table form. We also present the full set of figures for all the methods and levels when training with a few training samples per concept.

How fast can models adapt to unseen concepts. For completeness, we present the scores of all the models for $N = \{1, 2, 4, 8, 16, 32, 64, 128, \text{All}\}$ on IN-1K and $L_{1/2/3/4/5}$ in Fig. B.1 (raw scores) and Fig. B.2 (relative scores). These results, grouped by levels (*i.e.*, for IN-1K and for $L_{1/2/3/4/5}$ separately) are also presented in Tabs. B.1 to B.6 respectively. These additional results complement Sec. 3.4.2.

Generalization to unseen concepts. To access the raw numbers of the results discussed in Sec. 3.4.2, we refer the reader to Tabs. B.1 to B.6 and the $N = \text{All}$ columns, which correspond to the scores shown in Fig. 3.4(a).

ResNet	Transformer	NAS & Other
● ResNet50 (23.5M) ■ *a*-ResNet152 (58.1M)	▲ *a*-T2T-ViT-t-14 (21.1M) ▶ *a*-DeiT-S (21.7M) ◀ *a*-DeiT-S-distilled (21.7M) ▼ *a*-DeiT-B-distilled (86.1M)	★ *a*-Inception-v3 (25.1M) ✚ *a*-EfficientNet-B1 (6.5M) ✖ *a*-EfficientNet-B4 (17.5M) ◆ *a*-NAT-M4 (7.6M) ⧫ *a*-VGG19 (139.6M)

Self-Supervision		Web data	Regularization	
■ *s*-DINO	★ *s*-SimCLR-v2	■ *d*-Semi-Sup	■ *r*-ReLabel	▼ *r*-Adv-Robust
▲ *s*-SwAV	✚ *s*-MoCo-v2	▲ *d*-Semi-Weakly-Sup	▲ *r*-CutMix	★ *r*-MEAL-v2
▶ *s*-BarlowTwins	✖ *s*-MoCHi	▶ *d*-MoPro	▶ *r*-MixUp	
◀ *s*-OBoW	◆ *s*-CompReSS	◀ *d*-CLIP	◀ *r*-Manifold-MixUp	
▼ *s*-BYOL	⧫ *s*-InfoMin			

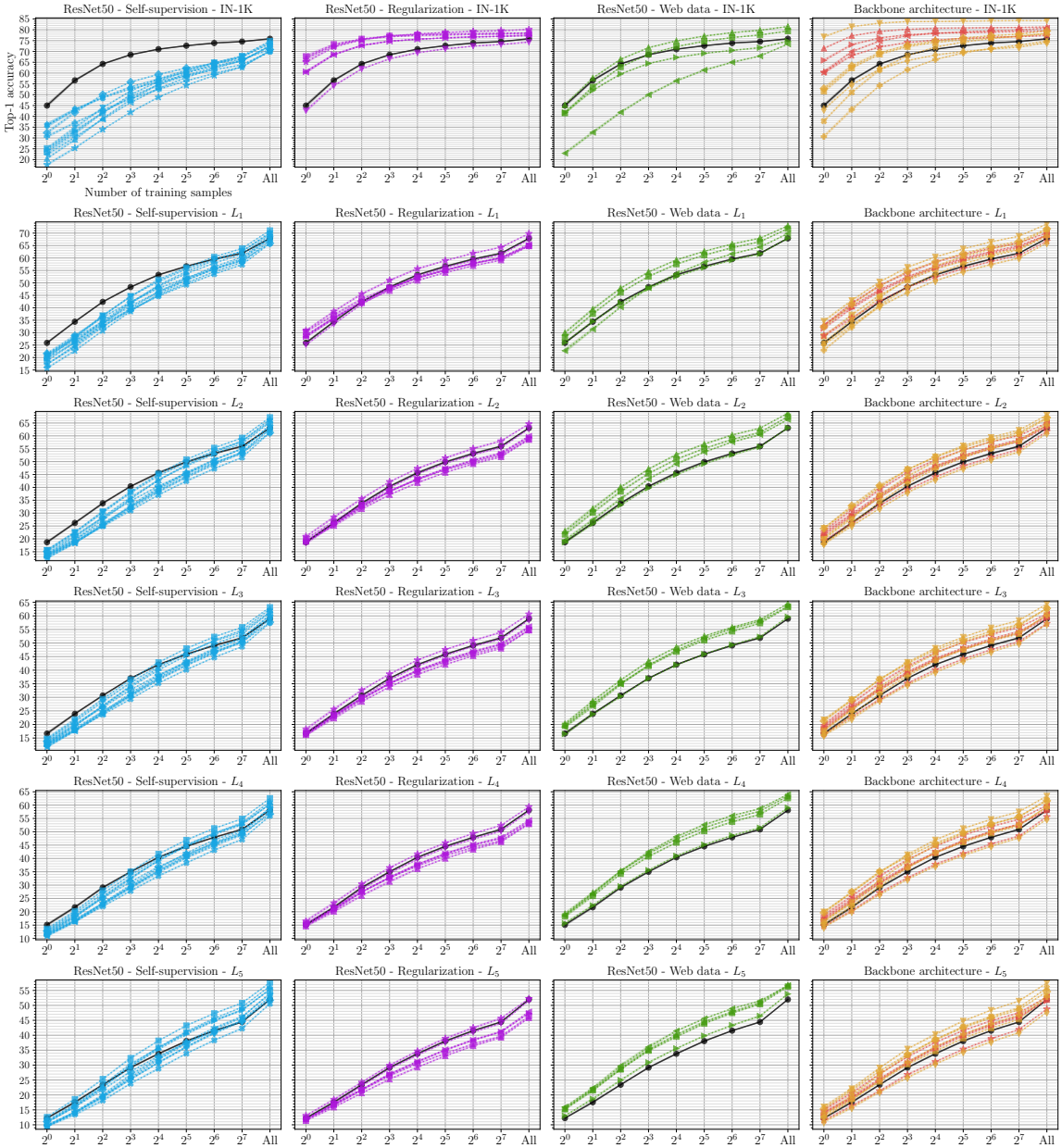


Figure B.1: Few-shot linear classification on ImageNet-CoG. Top-1 accuracy for each method using logistic regression classifiers. We train them on pre-extracted features for the concepts in IN-1K and our generalization levels ($L_{1/2/3/4/5}$), with a few training samples per concept, *i.e.*, $N = \{1, 2, 4, 8, 16, 32, 64, 128\}$. "All", the performance when all the samples are used, is also shown for reference.

Model	N-shots 2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	45.0 +- 0.7	56.6 +- 0.4	64.2 +- 0.2	68.5 +- 0.1	71.0 +- 0.0	72.6 +- 0.1	73.9 +- 0.1	74.6 +- 0.1	75.8 +- 0.0
a-ResNet-152	51.5 +- 0.7	62.3 +- 0.3	68.7 +- 0.1	72.2 +- 0.1	74.2 +- 0.1	75.4 +- 0.1	76.3 +- 0.1	77.0 +- 0.1	78.1 +- 0.1
a-T2T-ViTt-14	71.4 +- 0.3	77.2 +- 0.1	79.2 +- 0.1	80.2 +- 0.1	80.7 +- 0.1	80.8 +- 0.0	80.9 +- 0.0	81.1 +- 0.1	81.3 +- 0.0
a-DeiT-S	65.9 +- 0.5	73.1 +- 0.3	76.2 +- 0.2	77.7 +- 0.0	78.4 +- 0.1	78.6 +- 0.1	78.9 +- 0.0	79.1 +- 0.1	79.6 +- 0.0
a-DeiT-S distilled	60.4 +- 0.7	70.0 +- 0.2	74.8 +- 0.2	77.2 +- 0.1	78.4 +- 0.2	79.0 +- 0.1	79.5 +- 0.0	79.9 +- 0.1	80.8 +- 0.0
a-DeiT-B distilled	76.8 +- 0.3	81.4 +- 0.1	83.1 +- 0.2	83.8 +- 0.1	83.8 +- 0.0	84.0 +- 0.0	84.1 +- 0.1	84.2 +- 0.0	84.2 +- 0.0
a-Inception-v3	60.3 +- 0.7	68.1 +- 0.2	72.0 +- 0.2	74.1 +- 0.1	75.2 +- 0.1	76.0 +- 0.1	76.5 +- 0.1	76.8 +- 0.1	77.4 +- 0.0
a-EfficientNet-B1	30.7 +- 0.6	43.2 +- 0.6	54.2 +- 0.1	61.6 +- 0.2	66.2 +- 0.1	69.3 +- 0.2	71.4 +- 0.1	72.9 +- 0.1	74.7 +- 0.3
a-EfficientNet-B4	37.8 +- 0.3	51.1 +- 0.4	61.5 +- 0.5	68.1 +- 0.1	72.0 +- 0.1	74.3 +- 0.1	76.0 +- 0.1	77.1 +- 0.1	78.4 +- 0.1
a-NAT-M4	52.8 +- 0.7	63.3 +- 0.5	69.4 +- 0.2	72.8 +- 0.2	74.8 +- 0.1	76.2 +- 0.1	77.3 +- 0.1	78.0 +- 0.0	79.5 +- 0.1
a-VGG19	43.2 +- 0.4	54.6 +- 0.1	61.8 +- 0.2	65.8 +- 0.1	68.3 +- 0.1	69.9 +- 0.1	71.1 +- 0.1	72.0 +- 0.1	74.1 +- 0.1
s-DINO	23.6 +- 0.5	32.2 +- 0.7	41.6 +- 0.2	49.9 +- 0.2	56.2 +- 0.3	60.9 +- 0.1	64.7 +- 0.1	67.9 +- 0.2	74.8 +- 0.0
s-SwAV	20.5 +- 0.3	29.3 +- 0.4	39.0 +- 0.1	47.7 +- 0.1	54.8 +- 0.3	60.0 +- 0.1	64.1 +- 0.1	67.5 +- 0.1	74.3 +- 0.0
s-BarlowTwins	24.7 +- 0.6	33.3 +- 0.6	41.7 +- 0.2	49.0 +- 0.2	54.6 +- 0.2	59.0 +- 0.1	62.7 +- 0.1	65.7 +- 0.1	72.3 +- 0.0
s-OBoW	22.4 +- 0.6	30.4 +- 0.3	38.7 +- 0.2	46.4 +- 0.2	52.8 +- 0.2	58.0 +- 0.1	62.2 +- 0.1	65.7 +- 0.1	72.7 +- 0.0
s-BYOL	25.2 +- 0.5	34.7 +- 0.6	44.0 +- 0.2	51.7 +- 0.2	57.2 +- 0.2	61.4 +- 0.1	64.8 +- 0.2	67.5 +- 0.1	73.5 +- 0.0
s-SimCLR-v2	17.7 +- 0.5	25.3 +- 0.3	33.9 +- 0.1	41.9 +- 0.2	48.8 +- 0.2	54.3 +- 0.1	58.9 +- 0.1	62.8 +- 0.0	70.5 +- 0.0
s-MoCo-v2	30.6 +- 0.6	37.0 +- 0.3	43.0 +- 0.1	48.0 +- 0.2	52.5 +- 0.3	56.4 +- 0.2	59.8 +- 0.1	62.9 +- 0.2	70.1 +- 0.1
s-MoCHi	35.8 +- 0.9	43.1 +- 0.5	48.5 +- 0.2	52.7 +- 0.1	55.9 +- 0.3	58.9 +- 0.2	61.4 +- 0.1	63.9 +- 0.1	69.9 +- 0.1
s-CompReSS	32.4 +- 0.6	41.8 +- 0.5	50.1 +- 0.1	55.8 +- 0.1	59.4 +- 0.2	62.3 +- 0.2	64.5 +- 0.1	66.4 +- 0.1	70.9 +- 0.0
s-InfoMin	35.9 +- 0.8	43.1 +- 0.4	48.8 +- 0.1	53.6 +- 0.2	57.2 +- 0.3	60.4 +- 0.1	63.3 +- 0.1	65.9 +- 0.1	72.5 +- 0.0
d-Semi-Sup.	41.5 +- 0.6	53.6 +- 0.4	62.8 +- 0.1	68.7 +- 0.1	72.2 +- 0.2	74.7 +- 0.1	76.4 +- 0.1	77.6 +- 0.1	79.4 +- 0.0
d-Semi-Weakly-Sup.	45.2 +- 0.5	57.7 +- 0.2	66.4 +- 0.2	71.7 +- 0.1	74.9 +- 0.1	77.1 +- 0.2	78.7 +- 0.1	79.8 +- 0.1	81.5 +- 0.0
d-MoPro	41.8 +- 0.5	52.0 +- 0.2	59.6 +- 0.1	64.4 +- 0.1	67.2 +- 0.1	69.1 +- 0.1	70.5 +- 0.1	71.7 +- 0.1	74.7 +- 0.0
d-CLIP	22.9 +- 0.5	32.6 +- 0.5	41.9 +- 0.4	49.9 +- 0.3	56.4 +- 0.3	61.4 +- 0.3	65.0 +- 0.1	67.9 +- 0.1	73.4 +- 0.0
r-ReLabel	67.7 +- 0.8	73.3 +- 0.2	75.8 +- 0.0	77.2 +- 0.1	77.8 +- 0.1	78.1 +- 0.1	78.2 +- 0.1	78.4 +- 0.0	78.6 +- 0.0
r-CutMix	66.8 +- 0.5	72.7 +- 0.2	75.6 +- 0.1	76.8 +- 0.1	77.4 +- 0.1	77.6 +- 0.1	77.9 +- 0.1	78.0 +- 0.1	78.3 +- 0.0
r-Mixup	60.6 +- 0.4	68.6 +- 0.3	72.8 +- 0.2	74.7 +- 0.1	75.6 +- 0.1	76.2 +- 0.0	76.7 +- 0.0	76.9 +- 0.0	77.3 +- 0.0
r-Manifold Mixup	60.5 +- 0.5	68.5 +- 0.1	72.7 +- 0.2	74.7 +- 0.1	75.7 +- 0.0	76.3 +- 0.1	76.7 +- 0.1	77.0 +- 0.1	77.7 +- 0.0
r-AdvRobust	42.6 +- 0.7	54.1 +- 0.2	61.9 +- 0.1	66.6 +- 0.1	69.4 +- 0.1	71.2 +- 0.1	72.4 +- 0.1	73.2 +- 0.1	74.3 +- 0.1
r-MEAL-v2	65.1 +- 0.5	72.1 +- 0.2	75.4 +- 0.1	77.2 +- 0.2	78.1 +- 0.1	78.8 +- 0.2	79.3 +- 0.1	79.6 +- 0.1	80.1 +- 0.0

Table B.1: Top-1 accuracies obtained by linear classifiers on IN-1K. Table view corresponding to the 1st row in Fig. B.1.

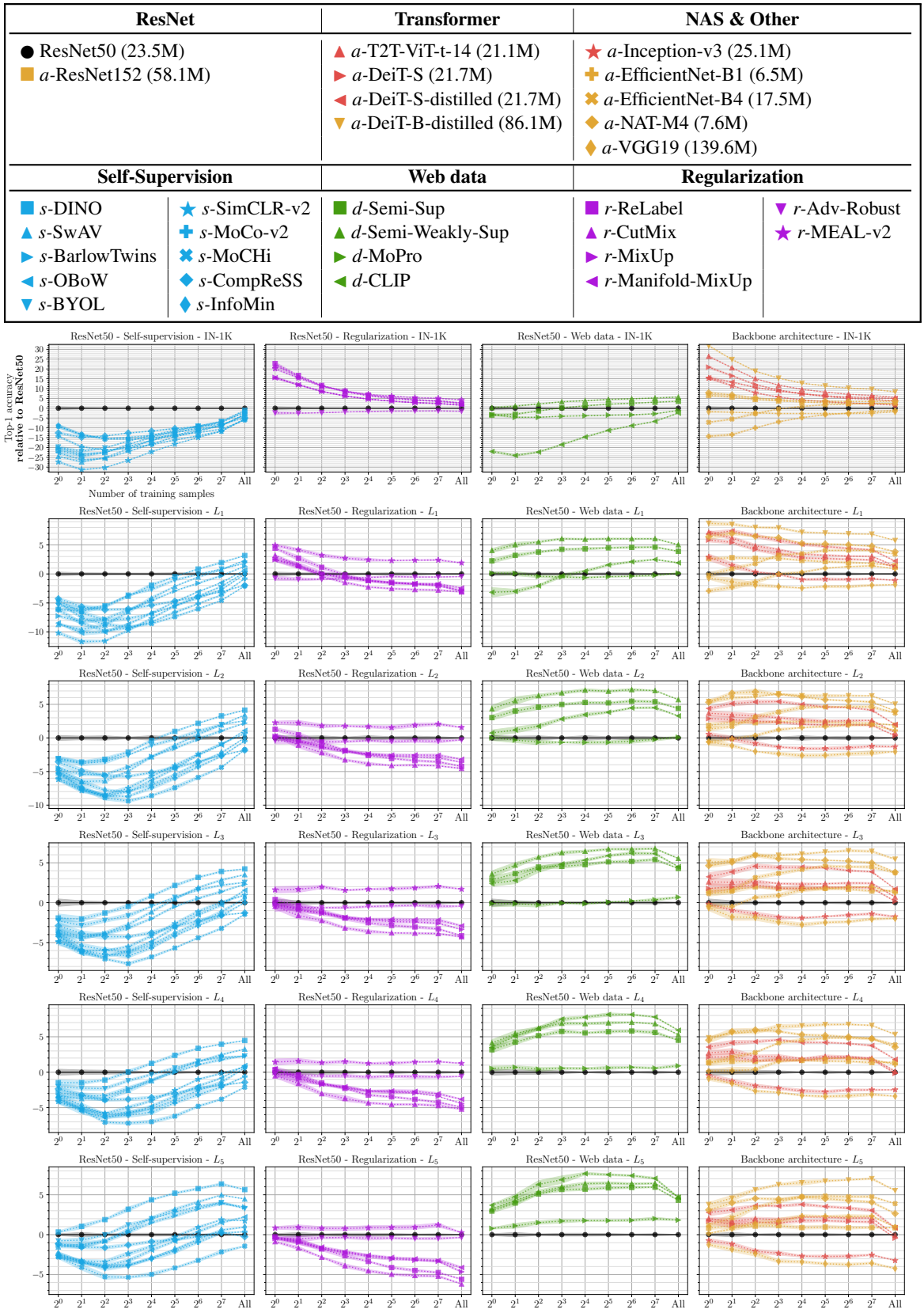


Figure B.2: Relative few-shot linear classification on ImageNet-CoG. The scores shown in Fig. B.1 from a different perspective: all scores are **relative to ResNet50**.

Model	N-shots 2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	25.9 +- 0.3	34.4 +- 0.2	42.4 +- 0.2	48.3 +- 0.1	53.2 +- 0.2	56.7 +- 0.2	59.6 +- 0.2	61.9 +- 0.1	67.9 +- 0.1
a-ResNet-152	28.5 +- 0.2	37.2 +- 0.3	45.1 +- 0.2	51.0 +- 0.2	55.6 +- 0.1	58.8 +- 0.2	61.5 +- 0.2	63.7 +- 0.1	69.3 +- 0.0
a-T2T-ViTt-14	33.1 +- 0.3	40.6 +- 0.3	47.2 +- 0.3	52.5 +- 0.2	56.6 +- 0.1	59.9 +- 0.1	62.6 +- 0.1	64.9 +- 0.1	69.2 +- 0.0
a-DeiT-S	31.7 +- 0.3	39.8 +- 0.3	46.6 +- 0.2	51.9 +- 0.3	56.0 +- 0.2	59.3 +- 0.2	62.1 +- 0.1	64.3 +- 0.1	69.0 +- 0.0
a-DeiT-S distilled	32.8 +- 0.4	41.8 +- 0.3	48.9 +- 0.3	54.1 +- 0.3	58.4 +- 0.2	61.4 +- 0.2	64.0 +- 0.2	66.1 +- 0.1	70.1 +- 0.1
a-DeiT-B distilled	34.6 +- 0.4	42.9 +- 0.2	50.4 +- 0.1	56.3 +- 0.2	60.4 +- 0.1	63.7 +- 0.1	66.5 +- 0.1	68.8 +- 0.0	73.7 +- 0.0
a-Inception-v3	28.8 +- 0.6	35.9 +- 0.5	42.7 +- 0.2	48.3 +- 0.1	52.3 +- 0.2	55.8 +- 0.2	58.6 +- 0.2	61.1 +- 0.1	66.8 +- 0.0
a-EfficientNet-B1	23.0 +- 0.3	32.1 +- 0.4	41.0 +- 0.4	48.3 +- 0.2	53.5 +- 0.2	57.6 +- 0.1	60.8 +- 0.1	63.3 +- 0.1	68.8 +- 0.0
a-EfficientNet-B4	25.7 +- 0.4	35.3 +- 0.3	44.1 +- 0.2	51.6 +- 0.2	56.7 +- 0.1	60.7 +- 0.2	63.6 +- 0.2	65.9 +- 0.1	71.3 +- 0.0
a-NAT-M4	32.3 +- 0.5	41.4 +- 0.3	48.9 +- 0.1	54.6 +- 0.1	58.5 +- 0.1	61.8 +- 0.2	64.6 +- 0.1	66.7 +- 0.1	71.7 +- 0.0
a-VGG19	25.2 +- 0.2	33.0 +- 0.7	40.5 +- 0.3	46.2 +- 0.2	50.8 +- 0.1	54.5 +- 0.2	57.4 +- 0.2	60.0 +- 0.1	66.1 +- 0.0
s-DINO	19.8 +- 0.1	27.9 +- 0.3	36.7 +- 0.2	44.5 +- 0.2	51.3 +- 0.3	56.2 +- 0.1	60.4 +- 0.1	63.9 +- 0.1	71.1 +- 0.0
s-SwAV	17.2 +- 0.1	24.8 +- 0.4	33.6 +- 0.2	41.8 +- 0.3	49.1 +- 0.3	54.5 +- 0.1	59.0 +- 0.1	62.6 +- 0.1	70.2 +- 0.0
s-BarlowTwins	18.6 +- 0.2	26.2 +- 0.3	34.6 +- 0.3	42.1 +- 0.2	48.6 +- 0.2	53.7 +- 0.1	58.0 +- 0.1	61.6 +- 0.2	69.2 +- 0.0
s-OBoW	17.4 +- 0.1	24.2 +- 0.4	32.4 +- 0.2	39.7 +- 0.2	46.4 +- 0.2	51.8 +- 0.1	56.5 +- 0.1	60.4 +- 0.1	68.6 +- 0.0
s-BYOL	20.5 +- 0.5	28.3 +- 0.3	36.9 +- 0.2	44.7 +- 0.1	50.6 +- 0.2	55.6 +- 0.2	59.5 +- 0.1	62.8 +- 0.1	69.7 +- 0.0
s-SimCLR-v2	15.7 +- 0.2	22.7 +- 0.3	30.8 +- 0.1	38.6 +- 0.2	45.6 +- 0.2	51.1 +- 0.2	55.7 +- 0.1	59.8 +- 0.1	68.2 +- 0.0
s-MoCo-v2	19.7 +- 0.3	25.9 +- 0.3	32.6 +- 0.3	39.0 +- 0.2	45.0 +- 0.1	50.0 +- 0.2	54.4 +- 0.1	58.3 +- 0.1	67.1 +- 0.0
s-MoCHi	21.0 +- 0.2	26.9 +- 0.3	33.7 +- 0.3	39.2 +- 0.3	44.7 +- 0.1	49.3 +- 0.2	53.5 +- 0.1	57.3 +- 0.1	65.8 +- 0.0
s-CompReSS	21.6 +- 0.2	28.8 +- 0.4	36.2 +- 0.2	42.3 +- 0.2	47.8 +- 0.2	52.0 +- 0.1	55.7 +- 0.1	58.8 +- 0.1	65.9 +- 0.0
s-InfoMin	21.4 +- 0.1	27.7 +- 0.3	34.4 +- 0.4	40.6 +- 0.3	46.4 +- 0.1	51.2 +- 0.1	55.4 +- 0.2	59.2 +- 0.1	67.9 +- 0.1
d-Semi-Sup.	28.1 +- 0.4	37.6 +- 0.3	46.1 +- 0.2	52.6 +- 0.1	57.6 +- 0.2	61.1 +- 0.1	64.1 +- 0.2	66.5 +- 0.1	71.8 +- 0.0
d-Semi-Weakly-Sup.	30.0 +- 0.3	39.4 +- 0.4	47.9 +- 0.2	54.4 +- 0.1	59.2 +- 0.1	62.7 +- 0.1	65.6 +- 0.1	68.0 +- 0.1	72.9 +- 0.0
d-MoPro	26.0 +- 0.2	34.5 +- 0.4	42.0 +- 0.2	47.8 +- 0.2	52.6 +- 0.1	56.2 +- 0.1	59.2 +- 0.1	61.7 +- 0.2	68.0 +- 0.1
d-CLIP	22.7 +- 0.8	31.4 +- 0.4	40.3 +- 0.2	48.0 +- 0.3	53.7 +- 0.2	58.2 +- 0.2	61.7 +- 0.1	64.4 +- 0.1	69.8 +- 0.1
r-ReLabel	30.3 +- 0.2	37.1 +- 0.4	43.5 +- 0.3	48.1 +- 0.3	52.1 +- 0.1	55.2 +- 0.2	57.8 +- 0.1	59.9 +- 0.1	64.9 +- 0.1
r-CutMix	29.1 +- 0.1	35.8 +- 0.3	41.9 +- 0.1	46.8 +- 0.2	51.0 +- 0.1	54.1 +- 0.1	56.9 +- 0.1	59.1 +- 0.1	64.8 +- 0.1
r-Mixup	28.4 +- 0.3	35.9 +- 0.4	42.5 +- 0.3	47.8 +- 0.3	52.1 +- 0.1	55.1 +- 0.1	57.8 +- 0.1	60.0 +- 0.2	65.0 +- 0.1
r-Manifold Mixup	28.4 +- 0.3	35.6 +- 0.3	42.3 +- 0.2	47.6 +- 0.3	52.0 +- 0.2	55.0 +- 0.2	57.9 +- 0.2	60.2 +- 0.1	65.5 +- 0.1
r-AdvRobust	25.2 +- 0.4	33.5 +- 0.3	41.4 +- 0.1	47.6 +- 0.2	52.6 +- 0.1	56.2 +- 0.2	59.0 +- 0.1	61.4 +- 0.1	67.4 +- 0.0
r-MEAL-v2	30.8 +- 0.3	38.5 +- 0.3	45.6 +- 0.3	51.0 +- 0.2	55.7 +- 0.2	59.0 +- 0.1	61.9 +- 0.1	64.3 +- 0.1	69.8 +- 0.1

Table B.2: Top-1 accuracies obtained by linear classifiers on L_1. Table view corresponding to the 2nd row in Fig. B.1.

Model	N-shots 2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	18.7 +- 0.4	26.2 +- 0.3	33.8 +- 0.3	40.4 +- 0.1	45.6 +- 0.3	49.9 +- 0.2	53.2 +- 0.3	56.0 +- 0.2	63.1 +- 0.0
a-ResNet-152	20.5 +- 0.4	28.6 +- 0.2	36.1 +- 0.2	42.8 +- 0.1	48.1 +- 0.2	52.2 +- 0.2	55.4 +- 0.2	58.1 +- 0.1	65.0 +- 0.0
a-T2T-ViTt-14	22.4 +- 0.4	29.6 +- 0.4	37.1 +- 0.3	43.4 +- 0.2	48.3 +- 0.3	52.4 +- 0.2	55.8 +- 0.2	58.6 +- 0.1	63.9 +- 0.0
a-DeiT-S	21.6 +- 0.4	28.9 +- 0.4	36.3 +- 0.4	42.7 +- 0.1	47.7 +- 0.2	51.8 +- 0.1	55.1 +- 0.1	57.9 +- 0.1	63.4 +- 0.0
a-DeiT-S distilled	23.1 +- 0.3	31.3 +- 0.4	39.2 +- 0.3	45.8 +- 0.2	50.6 +- 0.2	54.5 +- 0.2	57.7 +- 0.1	60.1 +- 0.1	65.1 +- 0.0
a-DeiT-B distilled	24.1 +- 0.3	32.1 +- 0.3	39.9 +- 0.1	47.0 +- 0.0	51.9 +- 0.2	56.2 +- 0.2	59.5 +- 0.1	62.2 +- 0.1	68.1 +- 0.0
a-Inception-v3	19.2 +- 0.3	26.1 +- 0.4	33.0 +- 0.3	39.1 +- 0.1	44.1 +- 0.2	48.3 +- 0.2	51.7 +- 0.3	54.7 +- 0.1	61.8 +- 0.1
a-EfficientNet-B1	18.1 +- 0.4	26.0 +- 0.2	34.2 +- 0.2	41.6 +- 0.2	47.2 +- 0.1	51.6 +- 0.2	55.1 +- 0.1	57.9 +- 0.1	64.2 +- 0.1
a-EfficientNet-B4	20.0 +- 0.3	27.9 +- 0.2	36.9 +- 0.3	44.3 +- 0.3	50.2 +- 0.3	54.5 +- 0.1	57.8 +- 0.1	60.6 +- 0.2	66.8 +- 0.0
a-NAT-M4	24.0 +- 0.4	32.9 +- 0.2	40.7 +- 0.4	46.9 +- 0.1	51.8 +- 0.1	55.6 +- 0.1	58.7 +- 0.1	61.3 +- 0.1	67.1 +- 0.0
a-VGG19	18.1 +- 0.5	25.0 +- 0.2	31.8 +- 0.0	38.1 +- 0.2	43.0 +- 0.3	47.3 +- 0.3	50.8 +- 0.2	53.7 +- 0.3	61.0 +- 0.0
s-DINO	15.6 +- 0.4	22.6 +- 0.2	30.6 +- 0.3	38.3 +- 0.2	45.3 +- 0.2	50.9 +- 0.1	55.3 +- 0.1	59.2 +- 0.1	67.2 +- 0.0
s-SwAV	13.5 +- 0.4	19.7 +- 0.3	27.4 +- 0.3	35.5 +- 0.1	42.8 +- 0.2	48.8 +- 0.2	53.7 +- 0.1	57.9 +- 0.1	66.5 +- 0.0
s-BarlowTwins	14.2 +- 0.3	20.7 +- 0.2	28.4 +- 0.2	36.1 +- 0.2	42.9 +- 0.1	48.3 +- 0.1	53.0 +- 0.2	57.0 +- 0.1	65.3 +- 0.0
s-OBoW	12.8 +- 0.5	18.4 +- 0.2	25.2 +- 0.3	32.4 +- 0.2	39.3 +- 0.2	45.3 +- 0.2	50.6 +- 0.1	54.9 +- 0.1	64.2 +- 0.1
s-BYOL	15.5 +- 0.3	22.4 +- 0.4	30.3 +- 0.3	37.8 +- 0.1	44.5 +- 0.2	49.7 +- 0.3	54.1 +- 0.2	58.0 +- 0.2	65.6 +- 0.0
s-SimCLR-v2	12.5 +- 0.2	18.3 +- 0.3	25.3 +- 0.3	32.9 +- 0.3	39.9 +- 0.1	45.8 +- 0.2	50.8 +- 0.1	55.1 +- 0.1	63.7 +- 0.0
s-MoCo-v2	13.4 +- 0.4	18.8 +- 0.3	25.2 +- 0.2	31.7 +- 0.1	38.3 +- 0.3	43.9 +- 0.2	48.9 +- 0.2	53.2 +- 0.1	62.6 +- 0.1
s-MoCHi	13.7 +- 0.2	18.7 +- 0.3	24.9 +- 0.4	31.0 +- 0.2	37.0 +- 0.1	42.5 +- 0.3	47.3 +- 0.2	51.6 +- 0.1	61.3 +- 0.0
s-CompReSS	15.1 +- 0.4	21.3 +- 0.4	28.2 +- 0.2	34.7 +- 0.2	40.5 +- 0.2	45.4 +- 0.2	49.7 +- 0.1	53.4 +- 0.1	61.3 +- 0.0
s-InfoMin	14.1 +- 0.4	19.6 +- 0.4	26.0 +- 0.2	32.4 +- 0.1	38.7 +- 0.2	44.4 +- 0.2	49.4 +- 0.2	53.7 +- 0.1	63.1 +- 0.0
d-Semi-Sup.	21.7 +- 0.4	30.2 +- 0.4	38.4 +- 0.3	45.4 +- 0.1	50.9 +- 0.2	55.1 +- 0.1	58.6 +- 0.2	61.4 +- 0.2	67.5 +- 0.0
d-Semi-Weakly-Sup.	23.0 +- 0.5	31.7 +- 0.6	40.1 +- 0.2	47.1 +- 0.2	52.8 +- 0.2	56.8 +- 0.2	60.3 +- 0.2	63.0 +- 0.1	68.8 +- 0.0
d-MoPro	18.7 +- 0.4	25.9 +- 0.4	33.2 +- 0.4	39.7 +- 0.1	45.0 +- 0.2	49.2 +- 0.2	52.7 +- 0.3	55.7 +- 0.2	63.2 +- 0.2
d-CLIP	19.4 +- 0.5	27.4 +- 0.4	35.6 +- 0.3	43.2 +- 0.2	49.1 +- 0.3	53.7 +- 0.1	57.6 +- 0.1	60.5 +- 0.1	66.4 +- 0.0
r-ReLabel	20.0 +- 0.2	26.6 +- 0.4	33.1 +- 0.2	38.6 +- 0.1	43.0 +- 0.2	46.8 +- 0.1	49.8 +- 0.2	52.4 +- 0.1	58.8 +- 0.1
r-CutMix	19.0 +- 0.2	25.1 +- 0.4	31.6 +- 0.3	37.1 +- 0.1	41.8 +- 0.2	45.8 +- 0.1	49.1 +- 0.1	51.8 +- 0.0	58.5 +- 0.1
r-Mixup	18.9 +- 0.3	25.7 +- 0.4	32.5 +- 0.3	38.4 +- 0.1	43.2 +- 0.1	47.1 +- 0.1	50.3 +- 0.1	53.0 +- 0.2	59.5 +- 0.1
r-Manifold Mixup	19.0 +- 0.3	25.8 +- 0.2	32.5 +- 0.3	38.5 +- 0.2	43.3 +- 0.2	47.3 +- 0.2	50.6 +- 0.1	53.4 +- 0.2	59.9 +- 0.1
r-AdvRobust	18.3 +- 0.5	25.5 +- 0.2	33.2 +- 0.3	39.9 +- 0.1	45.3 +- 0.2	49.4 +- 0.2	52.7 +- 0.2	55.5 +- 0.3	62.8 +- 0.0
r-MEAL-v2	20.9 +- 0.2	28.4 +- 0.4	35.6 +- 0.3	42.2 +- 0.1	47.3 +- 0.2	51.4 +- 0.3	55.1 +- 0.3	58.0 +- 0.2	64.6 +- 0.1

Table B.3: Top-1 accuracies obtained by linear classifiers on L_2. Table view corresponding to the 3rd row in Fig. B.1.

	N-shots								
Model	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	16.7 +- 0.5	23.9 +- 0.2	30.7 +- 0.1	37.0 +- 0.1	42.0 +- 0.1	45.9 +- 0.1	49.1 +- 0.1	51.9 +- 0.2	59.0 +- 0.0
a-ResNet-152	18.1 +- 0.4	25.5 +- 0.4	32.7 +- 0.2	38.8 +- 0.2	43.8 +- 0.1	47.7 +- 0.2	50.8 +- 0.1	53.4 +- 0.1	60.5 +- 0.0
a-T2T-ViTt-14	19.3 +- 0.4	26.5 +- 0.4	33.3 +- 0.2	39.4 +- 0.1	44.4 +- 0.2	48.2 +- 0.1	51.6 +- 0.1	54.3 +- 0.1	59.7 +- 0.0
a-DeiT-S	18.4 +- 0.6	25.9 +- 0.3	32.9 +- 0.4	38.8 +- 0.2	43.8 +- 0.3	47.8 +- 0.1	51.0 +- 0.1	53.7 +- 0.1	59.1 +- 0.0
a-DeiT-S distilled	19.9 +- 0.5	27.8 +- 0.3	35.2 +- 0.3	41.4 +- 0.2	46.5 +- 0.2	50.2 +- 0.1	53.1 +- 0.1	55.8 +- 0.1	60.7 +- 0.0
a-DeiT-B distilled	21.7 +- 0.5	29.1 +- 0.2	36.6 +- 0.2	43.0 +- 0.1	48.1 +- 0.1	52.2 +- 0.1	55.6 +- 0.1	58.3 +- 0.1	64.4 +- 0.1
a-Inception-v3	16.5 +- 0.2	22.9 +- 0.2	29.2 +- 0.3	35.2 +- 0.2	40.1 +- 0.1	44.1 +- 0.1	47.5 +- 0.1	50.5 +- 0.1	57.2 +- 0.1
a-EfficientNet-B1	16.2 +- 0.5	23.4 +- 0.2	31.3 +- 0.2	37.9 +- 0.3	43.6 +- 0.2	47.6 +- 0.2	51.0 +- 0.1	53.9 +- 0.1	60.2 +- 0.1
a-EfficientNet-B4	17.8 +- 0.3	25.3 +- 0.3	33.8 +- 0.2	40.7 +- 0.2	46.2 +- 0.2	50.4 +- 0.2	53.6 +- 0.1	56.4 +- 0.2	62.7 +- 0.1
a-NAT-M4	21.3 +- 0.3	29.0 +- 0.3	36.6 +- 0.2	42.5 +- 0.2	47.5 +- 0.1	51.2 +- 0.1	54.3 +- 0.0	56.9 +- 0.1	62.8 +- 0.0
a-VGG19	16.0 +- 0.4	22.1 +- 0.4	28.8 +- 0.4	34.6 +- 0.3	39.3 +- 0.2	43.4 +- 0.2	46.7 +- 0.1	49.9 +- 0.2	57.0 +- 0.1
s-DINO	14.7 +- 0.5	21.9 +- 0.3	29.4 +- 0.2	36.6 +- 0.3	42.9 +- 0.1	48.0 +- 0.2	52.3 +- 0.1	55.8 +- 0.1	63.2 +- 0.0
s-SwAV	12.9 +- 0.5	19.4 +- 0.2	26.8 +- 0.2	34.3 +- 0.2	40.9 +- 0.1	46.3 +- 0.2	51.0 +- 0.1	54.7 +- 0.1	62.5 +- 0.0
s-BarlowTwins	13.2 +- 0.5	19.6 +- 0.3	26.8 +- 0.2	33.8 +- 0.2	40.0 +- 0.1	45.4 +- 0.1	49.6 +- 0.1	53.3 +- 0.1	61.3 +- 0.0
s-OBoW	11.8 +- 0.3	17.6 +- 0.2	23.9 +- 0.1	30.9 +- 0.3	37.2 +- 0.1	42.8 +- 0.2	47.6 +- 0.1	51.6 +- 0.2	60.5 +- 0.1
s-BYOL	14.4 +- 0.5	21.0 +- 0.3	28.4 +- 0.2	35.4 +- 0.3	41.6 +- 0.1	46.5 +- 0.1	50.7 +- 0.1	54.1 +- 0.1	61.5 +- 0.0
s-SimCLR-v2	11.8 +- 0.4	17.9 +- 0.1	24.7 +- 0.1	31.5 +- 0.2	37.9 +- 0.2	43.2 +- 0.1	48.0 +- 0.2	52.0 +- 0.1	59.9 +- 0.1
s-MoCo-v2	12.5 +- 0.4	17.9 +- 0.2	24.2 +- 0.1	30.3 +- 0.3	36.4 +- 0.2	41.7 +- 0.1	46.2 +- 0.1	50.3 +- 0.2	59.0 +- 0.0
s-MoCHi	12.6 +- 0.5	17.9 +- 0.3	23.6 +- 0.1	29.4 +- 0.2	35.2 +- 0.1	40.2 +- 0.1	44.7 +- 0.1	48.7 +- 0.1	57.5 +- 0.1
s-CompReSS	13.8 +- 0.4	20.1 +- 0.3	26.4 +- 0.4	32.7 +- 0.1	38.3 +- 0.2	42.9 +- 0.2	46.8 +- 0.2	50.3 +- 0.1	57.7 +- 0.0
s-InfoMin	12.9 +- 0.5	18.5 +- 0.3	24.7 +- 0.3	30.8 +- 0.2	36.8 +- 0.2	42.0 +- 0.2	46.6 +- 0.2	50.7 +- 0.2	59.5 +- 0.1
d-Semi-Sup.	19.5 +- 0.5	27.5 +- 0.2	35.1 +- 0.2	41.6 +- 0.3	46.8 +- 0.2	51.0 +- 0.1	54.3 +- 0.2	57.3 +- 0.1	63.3 +- 0.0
d-Semi-Weakly-Sup.	20.2 +- 0.4	28.6 +- 0.3	36.4 +- 0.1	43.3 +- 0.2	48.5 +- 0.1	52.6 +- 0.1	55.8 +- 0.1	58.7 +- 0.1	64.5 +- 0.0
d-MoPro	16.5 +- 0.4	23.8 +- 0.3	30.4 +- 0.3	37.0 +- 0.1	41.9 +- 0.1	45.9 +- 0.1	49.3 +- 0.0	52.3 +- 0.2	59.6 +- 0.1
d-CLIP	19.0 +- 0.3	26.7 +- 0.4	34.8 +- 0.4	41.9 +- 0.2	47.4 +- 0.2	51.7 +- 0.1	55.3 +- 0.2	58.0 +- 0.1	63.5 +- 0.0
r-ReLabel	17.0 +- 0.4	23.5 +- 0.4	29.6 +- 0.2	35.1 +- 0.3	39.5 +- 0.2	43.0 +- 0.2	46.1 +- 0.2	48.6 +- 0.1	54.8 +- 0.1
r-CutMix	16.2 +- 0.3	22.3 +- 0.4	28.4 +- 0.2	33.8 +- 0.1	38.4 +- 0.2	42.1 +- 0.0	45.3 +- 0.1	48.0 +- 0.1	54.7 +- 0.1
r-Mixup	16.5 +- 0.5	23.1 +- 0.2	29.3 +- 0.2	35.1 +- 0.2	39.9 +- 0.1	43.5 +- 0.2	46.7 +- 0.1	49.3 +- 0.1	55.6 +- 0.0
r-Manifold Mixup	16.6 +- 0.4	23.0 +- 0.3	29.4 +- 0.1	35.2 +- 0.1	40.0 +- 0.1	43.8 +- 0.2	47.0 +- 0.2	49.7 +- 0.2	56.1 +- 0.1
r-AdvRobust	16.2 +- 0.6	23.3 +- 0.3	30.0 +- 0.1	36.4 +- 0.1	41.5 +- 0.1	45.5 +- 0.1	48.8 +- 0.1	51.5 +- 0.2	58.5 +- 0.0
r-MEAL-v2	18.3 +- 0.4	25.6 +- 0.4	32.6 +- 0.2	38.6 +- 0.1	43.7 +- 0.1	47.6 +- 0.2	50.9 +- 0.2	53.9 +- 0.2	60.7 +- 0.0

Table B.4: Top-1 accuracies obtained by linear classifiers on L_3. Table view corresponding to the 4$^{\text{th}}$ row in Fig. B.1.

Model	N-shots 2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	15.1 +- 0.4	21.7 +- 0.3	29.1 +- 0.3	35.0 +- 0.2	40.4 +- 0.0	44.5 +- 0.2	47.9 +- 0.1	50.9 +- 0.2	58.1 +- 0.0
a-ResNet-152	16.5 +- 0.5	23.2 +- 0.3	30.6 +- 0.2	36.8 +- 0.1	42.1 +- 0.1	46.1 +- 0.1	49.5 +- 0.1	52.3 +- 0.1	59.5 +- 0.0
a-T2T-ViTt-14	17.8 +- 0.3	24.7 +- 0.4	31.3 +- 0.2	37.2 +- 0.3	42.3 +- 0.1	46.8 +- 0.2	50.1 +- 0.1	52.9 +- 0.1	58.2 +- 0.1
a-DeiT-S	17.2 +- 0.3	23.9 +- 0.5	30.7 +- 0.2	36.8 +- 0.3	42.1 +- 0.1	46.5 +- 0.2	50.0 +- 0.2	52.7 +- 0.1	57.9 +- 0.0
a-DeiT-S distilled	18.7 +- 0.4	25.8 +- 0.3	33.4 +- 0.2	39.6 +- 0.1	44.6 +- 0.1	48.7 +- 0.1	52.0 +- 0.1	54.7 +- 0.1	59.9 +- 0.0
a-DeiT-B distilled	20.0 +- 0.2	27.2 +- 0.1	34.8 +- 0.2	41.4 +- 0.3	47.0 +- 0.2	51.3 +- 0.2	54.7 +- 0.1	57.5 +- 0.1	63.4 +- 0.1
a-Inception-v3	14.7 +- 0.3	20.4 +- 0.3	27.2 +- 0.3	32.7 +- 0.2	37.8 +- 0.2	41.8 +- 0.2	45.4 +- 0.2	48.4 +- 0.1	55.6 +- 0.1
a-EfficientNet-B1	15.3 +- 0.2	21.9 +- 0.5	29.5 +- 0.2	36.5 +- 0.2	42.1 +- 0.1	46.5 +- 0.0	49.9 +- 0.1	52.7 +- 0.1	59.2 +- 0.1
a-EfficientNet-B4	16.4 +- 0.2	23.8 +- 0.4	31.9 +- 0.1	39.1 +- 0.1	45.0 +- 0.3	49.4 +- 0.1	52.8 +- 0.1	55.6 +- 0.2	62.0 +- 0.1
a-NAT-M4	19.6 +- 0.4	27.4 +- 0.4	35.1 +- 0.2	40.8 +- 0.1	45.7 +- 0.1	49.7 +- 0.1	52.9 +- 0.1	55.7 +- 0.1	61.6 +- 0.0
a-VGG19	14.2 +- 0.3	20.1 +- 0.2	26.4 +- 0.2	32.2 +- 0.1	37.2 +- 0.1	41.1 +- 0.2	44.6 +- 0.2	47.8 +- 0.2	54.7 +- 0.0
s-DINO	13.6 +- 0.4	20.2 +- 0.3	28.0 +- 0.1	35.0 +- 0.3	41.7 +- 0.1	46.9 +- 0.2	51.3 +- 0.1	54.9 +- 0.1	62.6 +- 0.0
s-SwAV	11.8 +- 0.4	18.0 +- 0.2	25.3 +- 0.2	32.5 +- 0.3	39.2 +- 0.2	44.8 +- 0.2	49.4 +- 0.1	53.4 +- 0.1	61.4 +- 0.0
s-BarlowTwins	12.2 +- 0.4	18.3 +- 0.5	25.8 +- 0.2	32.7 +- 0.3	39.3 +- 0.2	44.6 +- 0.2	49.0 +- 0.1	52.7 +- 0.2	60.4 +- 0.0
s-OBoW	11.0 +- 0.3	16.2 +- 0.4	22.7 +- 0.2	29.2 +- 0.3	35.6 +- 0.2	41.1 +- 0.2	46.0 +- 0.2	50.1 +- 0.1	59.0 +- 0.0
s-BYOL	13.1 +- 0.3	19.4 +- 0.3	26.8 +- 0.2	33.8 +- 0.2	40.1 +- 0.2	45.1 +- 0.2	49.5 +- 0.1	53.0 +- 0.2	60.4 +- 0.0
s-SimCLR-v2	11.0 +- 0.3	16.5 +- 0.3	23.3 +- 0.1	30.0 +- 0.3	36.4 +- 0.1	42.0 +- 0.1	46.9 +- 0.1	50.7 +- 0.1	58.9 +- 0.0
s-MoCo-v2	11.6 +- 0.2	16.6 +- 0.3	22.9 +- 0.3	28.9 +- 0.3	34.8 +- 0.3	40.2 +- 0.2	44.8 +- 0.3	48.8 +- 0.1	57.5 +- 0.0
s-MoCHi	11.4 +- 0.2	16.4 +- 0.2	22.0 +- 0.2	27.8 +- 0.2	33.4 +- 0.2	38.3 +- 0.2	43.1 +- 0.2	47.1 +- 0.1	56.0 +- 0.1
s-CompReSS	12.7 +- 0.3	18.5 +- 0.3	25.2 +- 0.2	31.2 +- 0.2	36.9 +- 0.2	41.5 +- 0.2	45.7 +- 0.2	49.1 +- 0.1	56.7 +- 0.0
s-InfoMin	11.9 +- 0.3	17.0 +- 0.3	23.2 +- 0.2	29.3 +- 0.2	35.3 +- 0.1	40.6 +- 0.1	45.4 +- 0.3	49.5 +- 0.2	58.7 +- 0.1
d-Semi-Sup.	18.3 +- 0.4	25.9 +- 0.3	34.2 +- 0.2	40.8 +- 0.3	46.0 +- 0.1	50.2 +- 0.2	53.7 +- 0.1	56.5 +- 0.2	62.6 +- 0.0
d-Semi-Weakly-Sup.	19.2 +- 0.5	27.1 +- 0.4	35.3 +- 0.2	41.9 +- 0.2	47.3 +- 0.2	51.5 +- 0.2	55.0 +- 0.1	57.7 +- 0.1	63.4 +- 0.0
d-MoPro	15.6 +- 0.3	22.4 +- 0.2	29.5 +- 0.3	35.6 +- 0.2	41.0 +- 0.1	45.2 +- 0.2	48.6 +- 0.1	51.5 +- 0.1	59.0 +- 0.2
d-CLIP	18.6 +- 0.4	26.8 +- 0.3	35.3 +- 0.3	42.5 +- 0.3	48.2 +- 0.2	52.7 +- 0.2	56.1 +- 0.1	58.7 +- 0.1	64.0 +- 0.0
r-ReLabel	15.5 +- 0.4	21.3 +- 0.3	27.6 +- 0.1	32.7 +- 0.1	37.2 +- 0.1	41.1 +- 0.1	44.1 +- 0.1	46.7 +- 0.1	53.2 +- 0.0
r-CutMix	14.6 +- 0.3	20.1 +- 0.4	26.1 +- 0.2	31.3 +- 0.4	36.1 +- 0.1	40.0 +- 0.1	43.4 +- 0.1	46.2 +- 0.2	52.9 +- 0.1
r-Mixup	15.1 +- 0.4	20.7 +- 0.4	27.3 +- 0.0	32.8 +- 0.3	37.7 +- 0.1	41.7 +- 0.3	45.0 +- 0.1	47.5 +- 0.0	53.7 +- 0.1
r-Manifold Mixup	15.6 +- 0.0	20.9 +- 0.4	27.4 +- 0.2	32.9 +- 0.2	37.8 +- 0.1	41.9 +- 0.2	45.2 +- 0.2	47.9 +- 0.2	54.3 +- 0.1
r-AdvRobust	14.8 +- 0.4	21.3 +- 0.4	28.5 +- 0.1	34.5 +- 0.2	39.7 +- 0.2	44.0 +- 0.2	47.3 +- 0.2	50.3 +- 0.2	57.5 +- 0.0
r-MEAL-v2	16.5 +- 0.4	23.3 +- 0.4	30.4 +- 0.2	36.5 +- 0.2	41.7 +- 0.1	45.8 +- 0.2	49.4 +- 0.1	52.4 +- 0.1	59.4 +- 0.0

Table B.5: Top-1 accuracies obtained by linear classifiers on L_4. Table view corresponding to the 5$^{\text{th}}$ row in Fig. B.1.

Model	N-shots 2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	All
ResNet50	12.2 +- 0.1	17.6 +- 0.3	23.4 +- 0.1	29.2 +- 0.2	33.8 +- 0.2	38.0 +- 0.1	41.5 +- 0.1	44.4 +- 0.1	52.0 +- 0.0
a-ResNet-152	13.1 +- 0.2	18.7 +- 0.3	24.5 +- 0.2	30.3 +- 0.3	34.7 +- 0.2	39.0 +- 0.2	42.4 +- 0.2	45.3 +- 0.1	52.8 +- 0.0
a-T2T-ViTt-14	14.2 +- 0.2	19.2 +- 0.4	25.4 +- 0.2	31.0 +- 0.2	36.0 +- 0.1	40.2 +- 0.1	43.6 +- 0.2	46.5 +- 0.1	51.9 +- 0.0
a-DeiT-S	13.9 +- 0.3	18.8 +- 0.4	25.0 +- 0.1	30.6 +- 0.1	35.5 +- 0.2	39.8 +- 0.1	43.2 +- 0.1	46.0 +- 0.1	51.5 +- 0.0
a-DeiT-S distilled	15.0 +- 0.4	20.6 +- 0.3	27.1 +- 0.1	32.7 +- 0.1	37.6 +- 0.2	41.6 +- 0.1	44.9 +- 0.2	47.5 +- 0.2	52.8 +- 0.1
a-DeiT-B distilled	16.0 +- 0.3	22.1 +- 0.4	29.1 +- 0.2	35.4 +- 0.1	40.3 +- 0.3	44.8 +- 0.1	48.4 +- 0.0	51.5 +- 0.1	57.5 +- 0.0
a-Inception-v3	11.5 +- 0.3	16.4 +- 0.2	21.4 +- 0.3	26.8 +- 0.2	31.1 +- 0.1	35.3 +- 0.2	38.8 +- 0.2	41.9 +- 0.1	48.7 +- 0.1
a-EfficientNet-B1	12.4 +- 0.1	18.3 +- 0.2	24.9 +- 0.3	30.8 +- 0.4	36.1 +- 0.2	40.4 +- 0.1	43.8 +- 0.2	46.8 +- 0.1	52.9 +- 0.0
a-EfficientNet-B4	13.4 +- 0.2	19.7 +- 0.5	26.8 +- 0.2	33.3 +- 0.2	38.6 +- 0.2	42.9 +- 0.2	46.3 +- 0.2	49.2 +- 0.1	55.9 +- 0.0
a-NAT-M4	15.3 +- 0.3	21.6 +- 0.2	28.0 +- 0.1	33.6 +- 0.2	38.5 +- 0.2	42.6 +- 0.2	45.8 +- 0.1	48.5 +- 0.1	54.6 +- 0.1
a-VGG19	11.0 +- 0.2	15.7 +- 0.3	21.1 +- 0.3	25.8 +- 0.2	30.4 +- 0.2	34.4 +- 0.2	37.8 +- 0.2	40.9 +- 0.1	47.7 +- 0.1
s-DINO	12.6 +- 0.1	18.6 +- 0.3	25.3 +- 0.1	32.3 +- 0.3	38.2 +- 0.1	43.2 +- 0.1	47.3 +- 0.1	50.8 +- 0.1	57.6 +- 0.0
s-SwAV	10.7 +- 0.1	16.1 +- 0.3	22.7 +- 0.1	29.7 +- 0.3	35.9 +- 0.2	41.2 +- 0.2	45.6 +- 0.1	49.4 +- 0.0	56.4 +- 0.0
s-BarlowTwins	10.9 +- 0.2	16.3 +- 0.3	22.6 +- 0.2	29.5 +- 0.2	35.3 +- 0.2	40.4 +- 0.2	44.8 +- 0.1	48.4 +- 0.2	55.4 +- 0.0
s-OBoW	9.4 +- 0.2	14.0 +- 0.1	19.6 +- 0.2	25.9 +- 0.2	31.9 +- 0.2	37.2 +- 0.1	41.9 +- 0.2	45.9 +- 0.1	54.2 +- 0.0
s-BYOL	11.5 +- 0.2	16.9 +- 0.3	23.4 +- 0.1	30.4 +- 0.2	35.8 +- 0.1	40.9 +- 0.1	45.1 +- 0.1	48.5 +- 0.1	55.2 +- 0.0
s-SimCLR-v2	9.5 +- 0.2	14.3 +- 0.2	20.0 +- 0.2	26.8 +- 0.3	32.3 +- 0.2	37.7 +- 0.2	42.5 +- 0.2	46.3 +- 0.1	53.7 +- 0.0
s-MoCo-v2	9.7 +- 0.3	14.1 +- 0.3	19.4 +- 0.2	25.3 +- 0.2	30.8 +- 0.2	36.0 +- 0.3	40.6 +- 0.2	44.6 +- 0.1	52.3 +- 0.0
s-MoCHi	9.4 +- 0.2	13.5 +- 0.4	18.2 +- 0.3	23.8 +- 0.1	28.8 +- 0.3	33.9 +- 0.1	38.3 +- 0.2	42.3 +- 0.2	50.5 +- 0.1
s-CompReSS	11.1 +- 0.2	16.0 +- 0.2	21.8 +- 0.2	27.9 +- 0.2	32.9 +- 0.2	37.4 +- 0.2	41.3 +- 0.2	44.7 +- 0.1	51.7 +- 0.0
s-InfoMin	9.9 +- 0.2	14.2 +- 0.1	19.3 +- 0.3	25.3 +- 0.2	30.8 +- 0.1	36.2 +- 0.1	40.9 +- 0.1	45.0 +- 0.1	52.9 +- 0.0
d-Semi-Sup.	15.2 +- 0.2	21.5 +- 0.3	28.5 +- 0.1	34.8 +- 0.2	39.5 +- 0.2	43.9 +- 0.1	47.5 +- 0.1	50.4 +- 0.1	56.3 +- 0.1
d-Semi-Weakly-Sup.	15.5 +- 0.1	21.8 +- 0.4	28.8 +- 0.3	35.2 +- 0.2	40.2 +- 0.2	44.5 +- 0.2	47.9 +- 0.1	50.9 +- 0.1	56.7 +- 0.0
d-MoPro	13.0 +- 0.1	18.7 +- 0.3	24.9 +- 0.2	30.9 +- 0.2	35.6 +- 0.1	39.8 +- 0.1	43.3 +- 0.1	46.4 +- 0.2	53.8 +- 0.1
d-CLIP	15.9 +- 0.1	22.3 +- 0.3	29.7 +- 0.5	36.1 +- 0.5	41.5 +- 0.2	45.6 +- 0.1	48.9 +- 0.1	51.5 +- 0.1	56.7 +- 0.0
r-ReLabel	12.0 +- 0.2	16.8 +- 0.4	21.6 +- 0.2	26.5 +- 0.3	30.4 +- 0.1	33.9 +- 0.1	37.0 +- 0.2	39.7 +- 0.1	46.4 +- 0.1
r-CutMix	11.4 +- 0.1	15.9 +- 0.2	20.6 +- 0.1	25.3 +- 0.2	29.3 +- 0.2	33.1 +- 0.1	36.4 +- 0.2	39.2 +- 0.1	45.8 +- 0.0
r-Mixup	11.8 +- 0.2	16.6 +- 0.2	21.7 +- 0.1	26.9 +- 0.2	31.0 +- 0.2	35.0 +- 0.1	38.3 +- 0.0	41.1 +- 0.1	47.3 +- 0.1
r-Manifold Mixup	11.8 +- 0.2	16.7 +- 0.2	21.9 +- 0.1	27.0 +- 0.1	31.2 +- 0.1	35.2 +- 0.1	38.5 +- 0.2	41.2 +- 0.2	47.9 +- 0.1
r-AdvRobust	11.8 +- 0.1	17.1 +- 0.4	23.0 +- 0.2	28.8 +- 0.1	33.4 +- 0.3	37.6 +- 0.1	41.0 +- 0.1	44.0 +- 0.2	51.6 +- 0.1
r-MEAL-v2	13.1 +- 0.2	18.5 +- 0.3	24.2 +- 0.2	29.9 +- 0.2	34.7 +- 0.3	38.9 +- 0.3	42.5 +- 0.3	45.6 +- 0.3	52.2 +- 0.0

Table B.6: Top-1 accuracies obtained by linear classifiers on L_5. Table view corresponding to the last row in Fig. B.1.

Appendix C

Hyper-parameters for training t-ReX models on IN-1K

Table C.1: Hyper-parameters for training our models with ResNet50 architecture on IN-1K. Hyper-parameters shared by all models are given on the top part while the ones specific to **t-ReX** and **t-ReX*** are shown on the bottom part.

Configuration	Value for all models	
Optimizer	SGD	
Base learning rate	0.1	
Learning rate rule	$0.1 \times \text{batch size}/256$	
Learning rate warmup	Linear, 10 epochs	
Learning rate decay rule	Cosine schedule	
Weight decay	0.0001	
Momentum	0.9	
Number of GPUs	4	
Batch size per GPU	64	
Batch size total	256	
Epochs	100	
Synchronized batch norms	✓	
Mixed precision	✓	
τ in Equations (4.1) and (4.2)	0.1	
Augmentation pipeline from	DINO	
Number of crops	$M_g = 1, M_l = 8$	
Global crop resolution	224	
Global crop scale range	(0.4, 1)	
Local crop resolution	96	
Local crop scale range	(0.05, 0.4)	
	Value for **t-ReX**	Value for **t-ReX***
Projector input ℓ_2-norm	✓	✓
Projector L	3	1
Projector d_h	2048	2048
Projector d_b	256	256
Global crop scale range	(0.25, 1)	(0.4, 1)
Local crop scale range	(0.05, 0.25)	(0.05, 0.4)
Memory bank size $\lvert\mathcal{Q}\rvert$	8192	8192
Loss function used for training	$\mathcal{L}_{\text{OCM}}$	$\mathcal{L}_{\text{OCM}}$

Appendix D

Results per dataset for t-ReX models

In Tab. D.1, we report Top-1 accuracy on 13 transfer dataset and on IN-1K, which constitute the training-versus-transfer performance plane investigated in Sec. 4.4. Results are obtained by linear logistic regression classifiers, for the previous state-of-the-art models as well as our best models shown in Fig. 4.7.

Table D.1: Top-1 linear logistic regression accuracy per dataset. Mean LO is average log odds computed over all transfer datasets (*i.e.* all datasets except IN-1K). In Sec. 4.4, we only plot IN-1K and Mean LO scores for each model. We repeat each evaluation 5 times with different seeds; variance is generally negligible.

Model	*IN1K*	*CoG L_1*	*CoG L_2*	*CoG L_3*	*CoG L_4*	*CoG L_5*	*Aircraft*	*Cars196*	*DTD*	*EuroSAT*	*Flowers*	*Pets*	*Food101*	*SUN397*	*Mean LO*
Previous SotA															
DINO	74.8	71.1	67.2	63.2	62.6	**57.6**	62.5	67.4	**77.7**	**97.7**	95.6	88.9	78.7	66.0	1.256
PAWS	76.4	71.2	67.3	63.1	62.1	56.6	63.2	71.6	76.2	96.9	95.8	91.2	77.5	65.4	1.256
SL-MLP	75.1	70.1	66.1	61.6	60.4	54.5	63.1	70.9	75.0	96.7	94.8	91.6	74.9	63.7	1.189
LOOK+*multi-crop*	78.0	70.2	65.9	61.7	60.4	54.7	62.4	71.1	73.5	96.3	94.9	93.3	75.1	64.1	1.195
SupCon	78.8	69.9	64.7	60.6	59.1	53.1	57.3	60.9	74.6	95.7	91.6	92.8	71.9	62.8	1.053
RSB-A1	79.8	69.9	65.0	60.9	59.3	52.8	47.1	54.0	73.9	95.7	88.7	93.1	71.2	63.3	0.978
Our models on the convex hull in Fig. 4.7															
t-ReX	78.0	72.0	**68.3**	**63.9**	**63.4**	57.2	**67.3**	**74.2**	**77.7**	97.5	**96.2**	92.6	**80.1**	66.7	**1.357**
t-ReX-OCM (L=2, $\mathcal{Q}$|=8K)	78.8	**72.3**	68.2	63.7	63.0	56.8	64.7	70.8	75.8	97.3	95.3	93.2	79.1	**66.9**	1.305
t-ReX-OCM (L=1, h_ψ, |$\mathcal{Q}$|=131K)	79.6	71.7	67.3	62.8	61.6	55.3	61.9	68.8	75.2	96.7	94.0	**93.6**	76.6	66.1	1.224
t-ReX$_1$ (ℓ_2, L=1, d_h=4096, d_b=256)	79.8	71.7	67.1	63.0	61.8	54.8	61.1	66.7	74.4	96.8	93.2	93.5	76.7	66.2	1.201
t-ReX$_1$ (ℓ_2, L=1, d_h=2048, d_b=256)	80.0	71.3	66.4	62.3	60.6	53.9	58.8	67.5	75.2	96.4	91.6	93.4	75.4	65.4	1.150
t-ReX*	**80.2**	70.7	66.0	61.5	59.8	53.4	55.5	64.7	73.2	96.2	90.1	93.0	73.2	64.8	1.078

Appendix E

Extended qualitative results for synthetic ImageNet clones

In this section, we provide additional qualitative results for the synthetic ImageNet clones generated in Chapter 5. First we show random images for *all* ImageNet-100 classes from three datasets: ImageNet-100-Val (real images) and two ImageNet-100-SD datasets generated by the prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b". Then we discuss in more detail several types of issues that we observed in these synthetic images. Unless otherwise stated, the guidance scale used is 7.5.

Qualitative results for *all* ImageNet-100 classes. In Fig. E.5, we show a few random images from each of the 100 classes in ImageNet-100, for three datasets: i) The real images from ImageNet-100, ii) synthetic images generated by a simple prompt, which is only composed of the name of the class, and iii) synthetic images generated with guidance scale equal to 2.0 and a prompt that enforces thoses classes to appear in diverse backgrounds to improve the diversity of generated images. From this exhaustive list, even with a few images per class, one can observe a number of issues around the semantics, diversity and domain of those images.

Showcasing domain and diversity issues. We also show extended results for three classes in order to illustrate issues related to the domain and diversity. Fig. E.4 compares generated images between two fine-grained classes of crabs, while Fig. E.3 shows many images from multiple different generated datasets for a single dog class. We discuss both figures in the next sections.

E.1 Semantic errors

From closely inspecting the generated images we can see that there exists two classes for which the prompt $p_c =$ "c" produces images of the wrong semantics: For the classes "papillon" and "wing", we see the generated images in the middle column of Fig. E.5 to be wrong due to *polysemy* associated with the class names. What is more, although not fully visible from the small set of images we show here, we saw that semantics are partially wrong for at

least the classes “green mamba”, “walking stick” and “iron”. For “green mamba”, although the synset refers to the snake species, there is a car model of the same name appearing in some of the generated images instead. For “walking stick”, the synset refers to the insect, while a subset of the generated images also contained walking sticks that are not insects.

As we discuss in the Chapter 5, appending the hypernym or definition of each synset seems to fix polysemy issues in many cases, including the ones mentioned above. However, we can see at least two cases where adding the hypernym in the prompt leads to worse results. According to WordNet Miller [1995], the hypernym for “shih-tzu” is “toy dog” something that results in dog-shaped toys in many of the generated images (see also Fig. E.3). Another example is the class “boathouse”, where appending the parent class “shed” leads to sheds that are not inside a body of water.

E.2 NSFW content

Another issue that was not very prominent, but still visible, even in the case of generic animal and object categories present in ImageNet-100, was the fact that some of the generated images contained NSFW (Not Suitable For Work) content in the form of nudity. The open-source code for Stable Diffusion comes with a highly selective safety module, that discards generated images that might contain NSFW content.[1] We disabled this module when generating images for the ImageNet synsets as we wanted to study the model as-is first, and to understand the problem.

We thoroughly inspected all classes of ImageNet-100 and observed minor NSFW issues with two of the classes: 1) The basic prompt for the class “sarong” led to a few images that had partial nudity. This effect was exaggerated when adding the description of the concept that reads “a loose skirt consisting of brightly colored fabric wrapped around the body; worn by both women and men in the South Pacific". It seems that words like “body” biases the image generation process towards more NSFW content. 2) Prompts for the class “ski mask” in combination with certain backgrounds from the Places dataset Zhou et al. [2017] also resulted in nudity. Overall, we want to emphasize that the Stable Diffusion models we tested were all highly susceptible to generate such content.

E.3 Misrepresentation of biodiversity

The degree of misrepresentation of biodiversity in the images generated from Stable Diffusion is very high. We partially showcase the issue in Fig. E.4 where we show many generated images for two fine-grained classes, *i.e.*, “rock crab” and “fiddler crab”.

[1] https://huggingface.co/CompVis/stable-diffusion-v1-4?text=Safety

"Rock crab" is defined in WordNet as "crab of eastern coast of North America", while the "fiddler crab" as a "burrowing crab of American coastal regions having one claw much enlarged in the male". The fact that the male fiddler crab has one claw much larger is a prominent theme when it comes to the real ImageNet-100 images shown on the right side of Fig. E.4a.

It does not take an expert ecologist to see that, although most of the generated images capture the coarser class "crab", the visual differences between the two sets of images, *e.g.*, in Fig. E.4b, are not focusing on the single enlarged claw for the fiddler crab case. What is more, the exhibited intra-class visual diversity, *i.e.*, crabs of different shapes and colors, seems to exceed a single species of crab.

This is just a single example, but from our inspection of many other fine-grained animal and fungi classes, we could see that this is not an isolated issue. On the contrary, it seems prominent across many fine-grained domains. One exception for the subset of ImageNet classes we delved into is dog breeds, possibly due to the sheer volume of dog images on the internet. It is however fair to say that the generated images highly misrepresent biodiversity.

It is worth noting that, as Luccioni and Rolnick discuss in their recent work Luccioni and Rolnick [2022], the ImageNet dataset itself contains a number of issues when it comes to the annotations of fine-grained classes of wild animals. They found that "many of the classes are ill-defined or overlapping, and that 12% of the images are incorrectly labeled, with some classes having $> 90\%$ of images incorrect". Although we did not conduct a similar experiment using experts, we expect similar statistics to be much higher for the images generated by Stable Diffusion.

E.4 Semantic issues arising with backgrounds

A common issue we observe when adding diverse backgrounds to class images is that a subset of the generated images do not really contain the object, and merely reflect the background scene. See for example the images in the first and last row, on the last column of Fig. E.4c, and a few more spread in that figure, or the background samples for class "reel" in Fig. E.5. This is to be expected given how a prompt like this is relying on the compositionality of the Stable Diffusion model.

What is really interesting is that in some cases the resulting images, although not containing an instance from the class, retains some of the object's shape or texture in the background. See for example a pedestal-looking table in Fig. E.4c for class "pedestal", a pirate themed bedroom for class "pirate", green shirts for "green mamba", or the red-ish produce stand for "red fox".

E.5 Issues with diversity

We observe issues with diversity for most of the classes when only the class name is used as the prompt, *e.g.*, in the middle set of results in Fig. E.5. This is also visible for the crab classes in Fig. E.4b, or the Shih-tzu class in Fig. E.2b, Fig. E.3a and Fig. E.3b. We see that such issues are partially solved when lowering the guidance scale and relying less to the prompt, or using backgrounds (*e.g.*, the right-most set of images in Fig. E.5). We expect more advanced prompt engineering to further increase diversity.

As expected, increasing diversity correlates with more semantic errors. We see that such issues appear far more frequently in the most diverse synthetic dataset, *i.e.*, as shown in the right-most set of images of Fig. E.5.

E.6 Non-natural images

Even from the very small random sample of generated images shown in the figures of this work, we see that there is a non-negligible percentage of the generated images that are non-natural. They can be illustrations, graphics images or even paintings. This is not necessarily undesirable and it can lead to models with higher robustness to related domain changes.

E.7 Varying the stable diffusion parameters

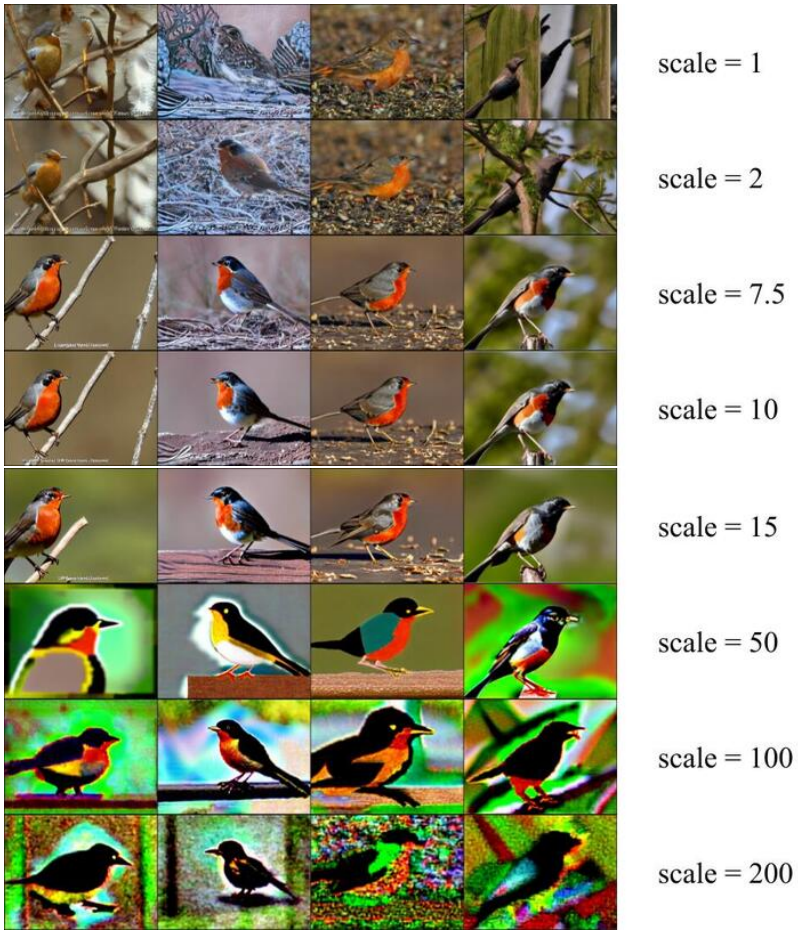
We identify two important parameters for Stable Diffusion, which affect the visual quality of generated images: The guidance scale and the number of diffusion steps. In Fig. E.1 we show several examples where we vary one of these two parameters. More specifically, we generate images for the ImageNet synset `n01558993` with class name "robin, American robin, Turdus migratorius", for the simplest case where the prompt is just the class name. We fix the seed to 1947262 and vary either the guidance scale or the number of diffusion steps.

Guidance Scale. From Fig. E.1a, we see that increasing the guidance scale coefficient over 10 starts giving hyper-realistic results. When the scale is under 2, we see that many details of the class are not really prominent.

Diffusion Steps. From Fig. E.1b, we see that, although with 5 steps the generated images still contain a lot of noise, running 25-50 steps is enough for fully-formed, sharp images to emerge. Since this is a parameter that linearly impacts generation time, increasing the number of steps further than 50 seems excessive.

Output Resolution. The resolution that was used during training of the Stable Diffusion models was (512×512).[2] We notice that if one deviates from this training resolution, generated results get worse. We chose to simply switch the aspect ratio to the one for the average ImageNet image and keep the long dimension to 512.

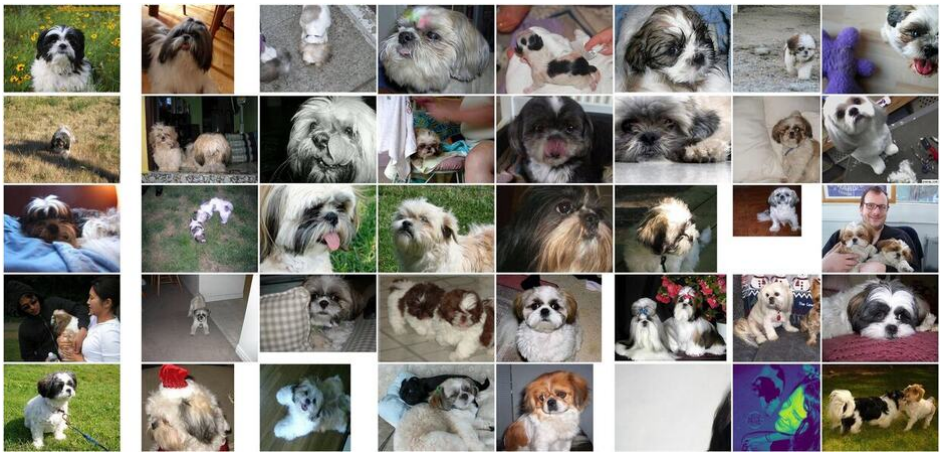
[2] https://github.com/CompVis/stable-diffusion



(a) Varying the guidance scale parameter (steps = 50)

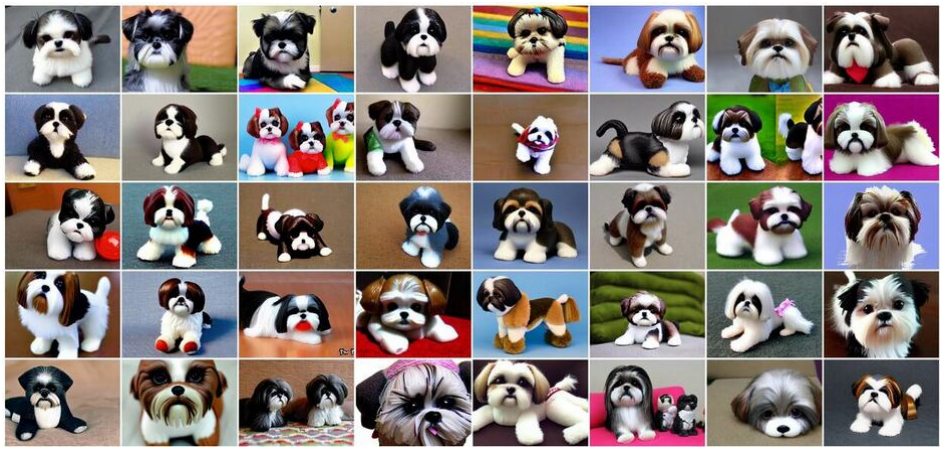
(b) Varying the number of diffusion steps (scale = 7.5)

Figure E.1: Qualitative results as we change the guidance scale parameter and the number of diffusion steps during Stable Diffusion generation. The seed is fixed to `1947262` and the prompt is "robin, American robin, Turdus migratorius". Unless otherwise stated the scale (resp. steps) parameters are set to 7.5 (resp. 50).



(a) Real images from ImageNet-1K for class "Shih-Tzu"

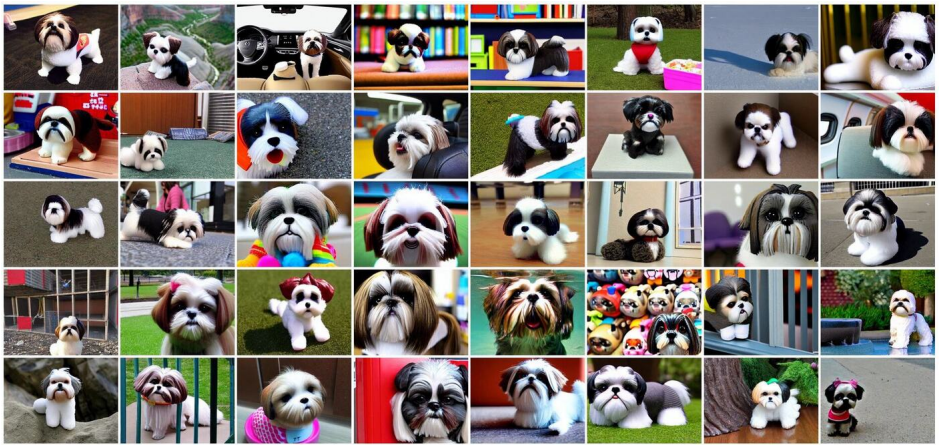
(b) Synthetic images with prompt p_c = "c" for class "Shih-Tzu"



(c) Synthetic images with prompt p_c = "c, h_c" for class "Shih-Tzu"

Figure E.2: Qualitative results for class "Shih-Tzu" to illustrate domain and diversity issues. Guidance scale is equal to 7.5.

(a) (cont.) Synthetic images with prompt $p_c =$ "c, d_c" for class "Shih-Tzu"

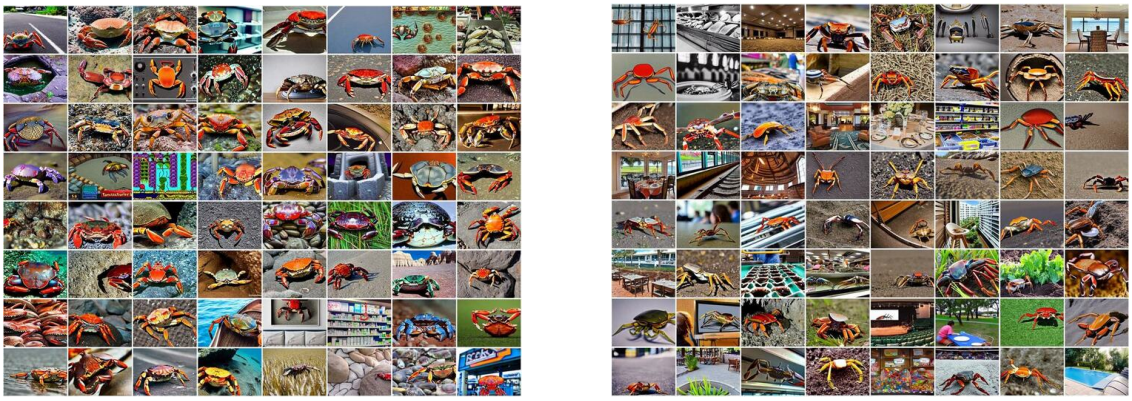


(b) Synthetic images with prompt $p_c =$ "c, h_c inside b"

Figure E.3: (cont.) Qualitative results for class "Shih-Tzu" to illustrate domain and diversity issues.

(a) Real images from ImageNet-1K for classes "Rock crab" (left) and "Fiddler crab" (right)

(b) Synthetic images with prompt p_c = "c" for classes "Rock crab" (left) and "Fiddler crab" (right)



(c) Synthetic images with prompt p_c = "c, h_c inside b" for classes "Rock crab" (left) and "Fiddler crab" (right)

Figure E.4: Qualitative results for classes "Rock crab" (left) and "Fiddler crab" (right), to illustrate issues around fine-grained and domain specific semantics. Guidance scale is equal to 7.5.

Figure E.5: Visualization of the 100 ImageNet-100 classes for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b".

Figure E.6: (cont.) Visualization of the 100 ImageNet-100 classes for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b".

Figure E.7: (cont.) Visualization of the images for the 100 ImageNet-100 classes in the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts p_c = "c" and p_c = "c, h_c inside b".

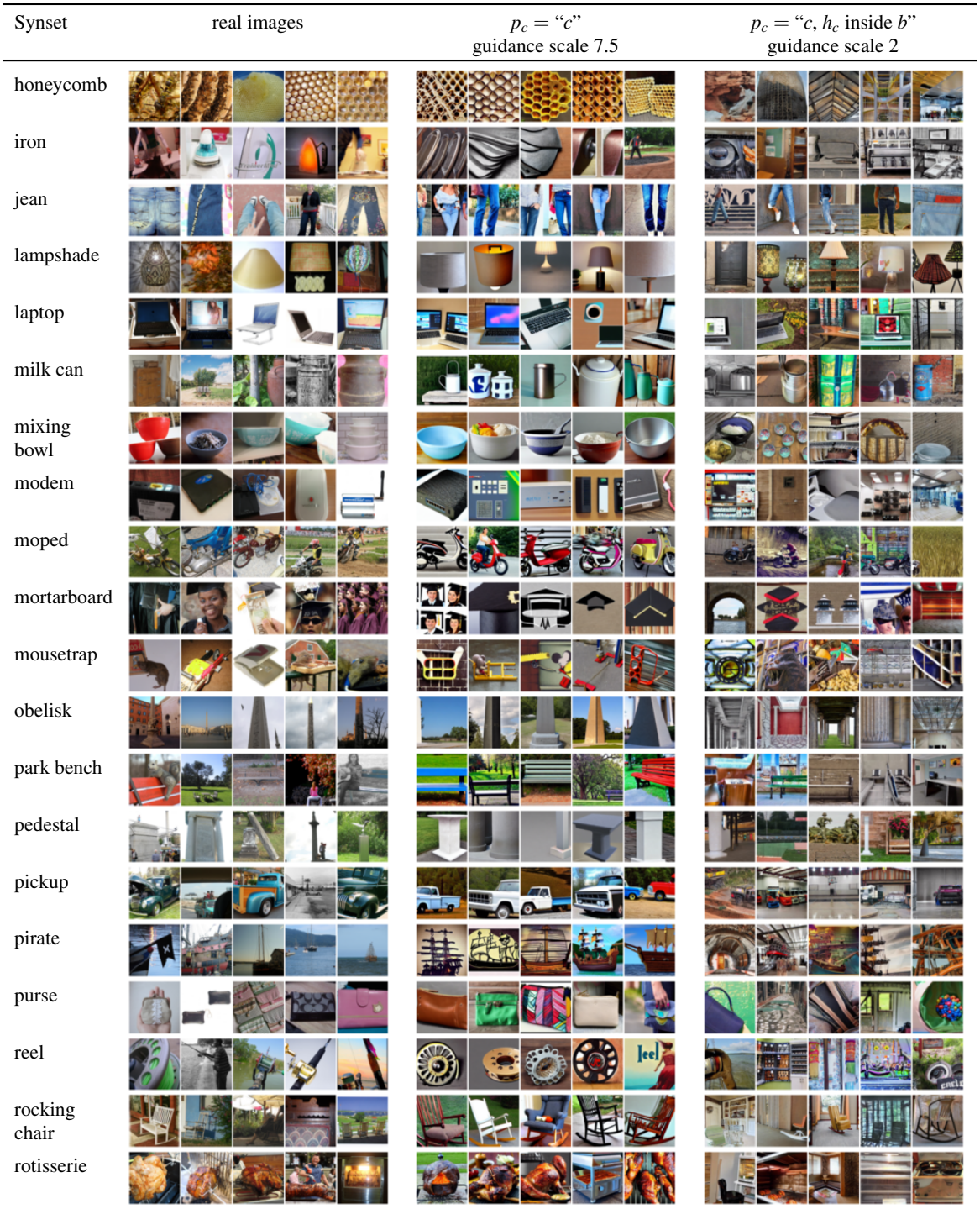


Figure E.8: (cont.) Visualization of the 100 ImageNet-100 classes for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts p_c = "c" and p_c = "c, h_c inside b".

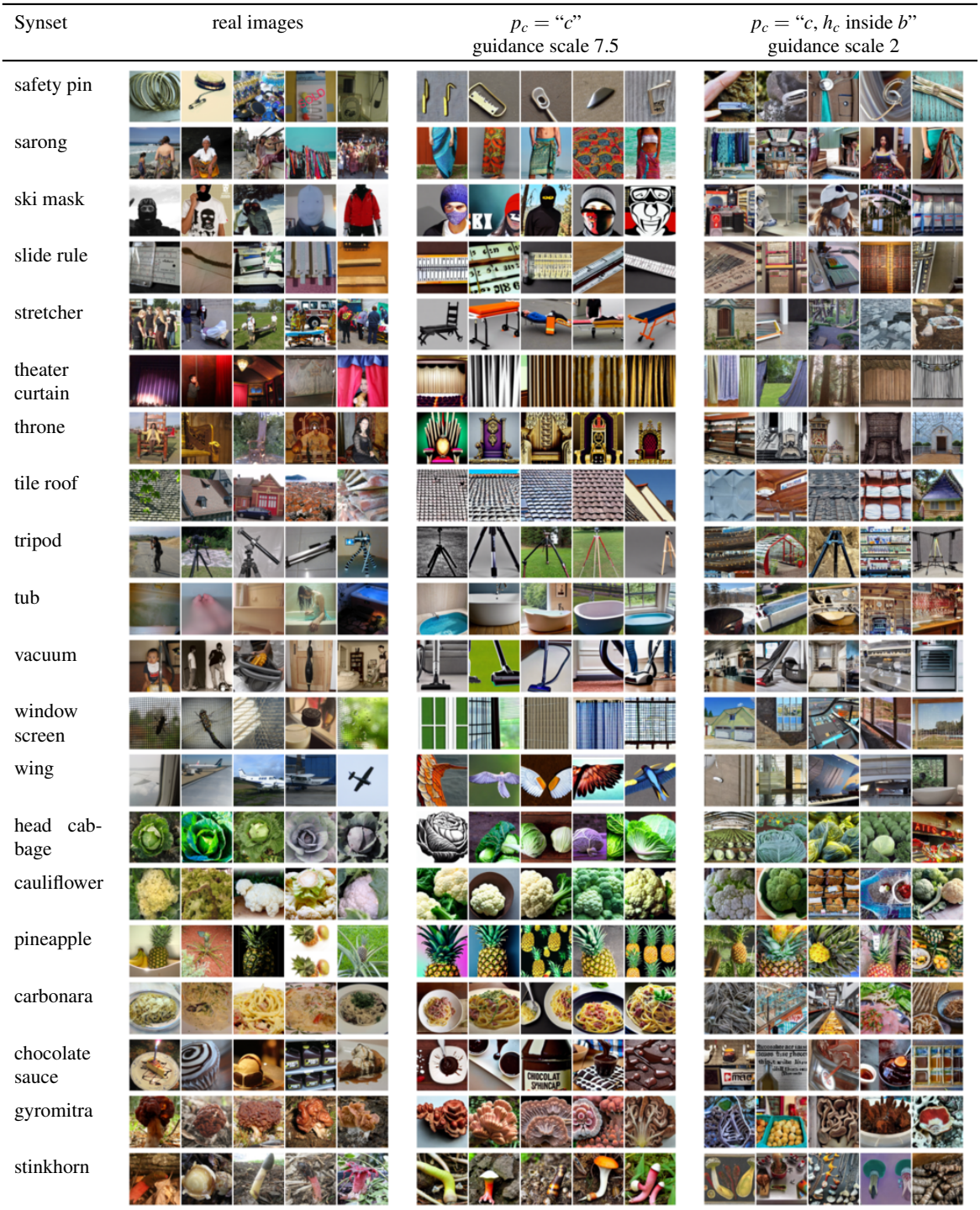


Figure E.9: (cont.) Visualization of the 100 ImageNet-100 classes for the three different datasets: ImageNet-100-Val (real) and two ImageNet-100-SD datasets created with prompts $p_c =$ "c" and $p_c =$ "c, h_c inside b".

Bibliography

Andrea Agostinelli, Jasper Uijlings, Thomas Mensink, and Vittorio Ferrari. Transferability metrics for selecting source model ensembles. In Proc. CVPR, 2022. (page 28.)

Osman Aka, Ken Burke, Alex Bauerle, Christina Greer, and Margaret Mitchell. Measuring model biases in the absence of ground truth. In AIES, 2021. (page 97.)

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In Proc. ICKDDM, 2019. (pages 42, 66, and 92.)

Alexander Amini, Igor Gilitschenski, Jacob Phillips, Julia Moseyko, Rohan Banerjee, Sertac Karaman, and Daniela Rus. Learning robust control policies for end-to-end autonomous driving from data-driven simulation. RAL, 2020. (page 24.)

Yuki M. Asano. Learning deep neural networks: Necessity and scope of prior knowledge, raw data, and labels. PhD thesis, University of Oxford, 2021. (page 23.)

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In Proc. ICLR, 2020. (page 23.)

Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. In Proc. ICCV, 2021. (pages xiv, 9, 20, 24, 58, 59, 60, 71, and 72.)

Yannis Avrithis and Yannis Kalantidis. Approximate Gaussian mixtures for large scale vocabularies. In Proc. ECCV, 2012. (page 15.)

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv:1607.06450, 2016. (page 17.)

Yajie Bao, Yang Li, Shao-Lun Huang, Lin Zhang, Lizhong Zheng, Amir Zamir, and Leonidas Guibas. An information-theoretic approach to transferability in task transfer learning. In Proc. ICIP, 2019. (page 28.)

Manel Baradad Jurjo, Jonas Wulff, Tongzhou Wang, Phillip Isola, and Antonio Torralba. Learning to see by looking at noise. In Proc. NeurIPS, 2021. (page 7.)

Jonathan Baxter. A model of inductive bias learning. JAIR, 2000. (page 27.)

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). CVIU, 110(3), 2008. (page 15.)

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. PAMI, 35(8), 2013. (page 13.)

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In Proc. ICML-W, 2012. (page 1.)

Tamara Berg and Alexander Berg. Finding iconic images. In Proc. CVPR-W, 2009. (page 36.)

Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. This dataset does not exist: training models from generated images. In ICASSP, 2020. (pages 7, 25, and 81.)

Lucas Beyer, Olivier J Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with ImageNet? arXiv:2006.07159, 2020. (page 21.)

Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain ViT baselines for ImageNet-1k. arXiv:2205.01580, 2022. (page 62.)

Federico Bianchi, Pratyusha Kalluri, Esin Durmus, Faisal Ladhak, Myra Cheng, Debora Nozza, Tatsunori Hashimoto, Dan Jurafsky, James Zou, and Aylin Caliskan. Easily accessible text-to-image generation amplifies demographic stereotypes at large scale. arXiv:2211.03759, 2022. (page 97.)

Steven Bird, Ewan Klein, and Edward Loper. Natural language processing with Python. O'Reilly, 2009. (page 37.)

Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. Multimodal datasets: Misogyny, pornography, and malignant stereotypes. arXiv:2110.01963, 2021. (page 97.)

Christopher M Bishop and Nasser M Nasrabadi. Pattern recognition and machine learning, volume 4. Springer, 2006. (page 16.)

Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In Proc. CVPR, 2008. (page 15.)

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. arXiv:2108.07258, 2021. (page 1.)

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – Mining discriminative components with random forests. In Proc. ECCV, 2014. (pages 66 and 92.)

Jules Bourcier, Gohar Dashyan, Jocelyn Chanussot, and Karteek Alahari. Evaluating the label efficiency of contrastive self-supervised learning for multi-resolution satellite imagery. In Image and Signal Processing for Remote Sensing, volume 12267, pages 152–161. SPIE, 2022a. (page 23.)

Jules Bourcier, Thomas Floquet, Gohar Dashyan, Tugdual Ceillier, Karteek Alahari, and Jocelyn Chanussot. Self-supervised pretraining on satellite imagery: A case study on label-efficient vehicle detection. arXiv:2210.11815, 2022b. (pages 23 and 102.)

Stevo Bozinovski. Reminder of the first paper on transfer learning in neural networks, 1976. Informatica, 44(3), 2020. (page 19.)

Stevo Bozinovski and Ante Fulgosi. The influence of pattern similarity and transfer learning upon training of a base perceptron b2. In Proc. Symposium Informatica, volume 3, 1976. (pages 19 and 27.)

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In Proc. ICLR, 2019. (pages 3, 6, 7, 23, 25, and 81.)

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In Proc. SIGKDD, 2006. (pages 20 and 82.)

Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based measures of lexical semantic relatedness. Computational Linguistics, 32(1), 2006. (page 38.)

Mateusz Budnik and Yannis Avrithis. Asymmetric metric learning for knowledge transfer. In Proc. CVPR, 2021. (page 20.)

Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. Information, 11(2), 2020. (page 17.)

Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. arXiv:2301.13188, 2023. (page 97.)

Mathilde Caron. Self-supervised learning of deep visual representations. PhD thesis, Université de Grenoble, 2021. (pages 19, 20, 22, and 23.)

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In Proc. ECCV, 2018. (pages 6, 17, and 23.)

Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised Pre-Training of Image Features on Non-Curated Data. In Proc. ICCV, 2019. (pages 27 and 35.)

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In Proc. NeurIPS, 2020. (pages xiii, 23, 24, 37, 44, 58, 59, 60, 61, 62, and 100.)

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In Proc. ICCV, 2021. (pages xiv, xviii, 6, 9, 19, 24, 44, 58, 59, 60, 62, 66, 67, 71, 72, 88, 89, 92, and 93.)

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In Proc. CVPR, 2022. (page 82.)

Lucy Chai, Jonas Wulff, and Phillip Isola. Using latent space regression to analyze and leverage compositionality in GANs. In Proc. ICLR, 2021a. (pages 25 and 82.)

Lucy Chai, Jun-Yan Zhu, Eli Shechtman, Phillip Isola, and Richard Zhang. Ensembling with deep generative views. In Proc. CVPR, 2021b. (pages 25 and 82.)

Ken Chatfield, Victor S Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In Proc. BMVC, 2011. (page 15.)

Hanting Chen, Yunhe Wang, Chang Xu, Zhaohui Yang, Chuanjian Liu, Boxin Shi, Chunjing Xu, Chao Xu, and Qi Tian. Data-free learning of student networks. In Proc. ICCV, 2019a. (page 83.)

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Proc. ICML, 2020a. (pages xi, xiii, 8, 17, 18, 19, 22, 23, 24, 27, 28, 32, 35, 44, 59, 60, 61, 63, and 100.)

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In Proc. NeurIPS, 2020b. (pages 37, 44, and 45.)

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In Proc. CVPR, 2021. (page 63.)

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. arXiv:2003.04297, 2020c. (pages 24, 44, and 63.)

Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In Proc. CVPR, 2019b. (pages 24 and 81.)

Yunqiang Chen, Xiang Sean Zhou, and Thomas S Huang. One-class SVM for learning in image retrieval. In Proc. ICIP, 2001. (page 79.)

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In Proc. CVPR, 2014. (pages xiii, 61, 66, and 92.)

Ramazan Gokberk Cinbis. Fisher kernel based models for image classification and object localization. PhD thesis, Université de Grenoble, 2014. (page 14.)

Gabriela Csurka, editor. Domain Adaptation in Computer Vision Applications. Advances in Computer Vision and Pattern Recognition. Springer, 2017. (pages 27 and 35.)

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In Proc. ECCV-W, 2004. (pages 1, 4, and 14.)

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In Proc. CVPR, 2019. (page 3.)

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In Proc. CVPR, 2005. (page 15.)

Celso M de Melo, Antonio Torralba, Leonidas Guibas, James DiCarlo, Rama Chellappa, and Jessica Hodgins. Next-generation deep learning based on simulators and synthetic data. TCS, 2021. (page 24.)

Hanneke EM Den Ouden, Peter Kok, and Floris P De Lange. How prediction errors shape perception, attention, and motivation. Frontiers in psychology, 3:548, 2012. (page 17.)

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In Proc. CVPR, 2009. (pages xi, xii, 3, 7, 16, 29, 33, 34, 37, 79, 84, and 99.)

Emily Denton, Alex Hanna, Razvan Amironesei, Andrew Smart, and Hilary Nicole. On the genealogy of machine learning datasets: A critical history of ImageNet. Big Data & Society, 2021. (page 97.)

Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In Proc. CVPR, 2021. (page 20.)

Thomas Deselaers and Vittorio Ferrari. Visual and semantic similarity in ImageNet. In Proc. CVPR, 2011. (pages 5, 32, 36, and 38.)

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. arXiv:1708.04552, 2017. (page 17.)

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In Proc. NeurIPS, 2021. (page 7.)

Carl Doersch, Abhinav Gupta, and Alexei Efros. Unsupervised visual representation learning by context prediction. In Proc. ICCV, 2015. (page 23.)

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In Proc. ICML, 2014. (pages 58 and 60.)

A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. PAMI, 38(9), 2016. (pages 6 and 23.)

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In Proc. ICCV, 2015. (pages 24 and 81.)

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In CORL, 2017. (page 24.)

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In Proc. ICLR, 2021. (page 16.)

Lisa Dunlap, Clara Mohri, Devin Guillory, Han Zhang, Trevor Darrell, Joseph E. Gonzalez, Aditi Raghunathan, and Anna Rohrbach. Using language to extend to unseen domains. In Proc. ICLR, 2023. (page 82.)

Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In Proc. ICCV, 2013. (page 82.)

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. JMLR, 20(1), 2019. (page 3.)

Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In Proc. CVPR, 2021. (pages 27 and 35.)

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In Proc. CVPR, 2021. (page 7.)

Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C Mozer. Head2toe: Utilizing intermediate representations for better transfer learning. In Proc. ICML, 2022. (pages 19 and 61.)

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. IJCV, 88, 2009. (pages 8, 27, 36, and 79.)

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In Proc. CVPR-W, 2004. (page 79.)

Yutong Feng, Jianwen Jiang, Mingqian Tang, Rong Jin, and Yue Gao. Rethinking supervised pre-training for better downstream transferring. In Proc. ICLR, 2022. (pages xiv, 6, 22, 59, 60, 61, 62, 70, 71, and 72.)

Andrea Frome, Greg Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In Proc. NeurIPS, 2013. (page 35.)

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, 36(4), 1980. (page 16.)

Jonas Geiping, Micah Goldblum, Gowthami Somepalli, Ravid Shwartz-Ziv, Tom Goldstein, and Andrew Gordon Wilson. How much data are augmentations worth? An investigation into scaling laws, invariance, and implicit regularization. In Proc. ICLR, 2023. (page 87.)

Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In Proc. NeurIPS, 2018. (pages 26 and 35.)

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In Proc. ICLR, 2018. (pages 3, 6, 19, 23, and 61.)

Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. Online bag-of-visual-words generation for unsupervised representation learning. In Proc. CVPR, 2021. (pages 23 and 44.)

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proc. CVPR, 2014. (pages 4 and 18.)

Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. GANalyze: Toward visual definitions of cognitive image properties. In Proc. ICCV, 2019. (page 25.)

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. In Proc. NeurIPS, 2004. (pages 22 and 60.)

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Proc. NeurIPS, 2014. (page 29.)

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016. (pages xi, 1, 2, 3, and 16.)

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. Communications of the ACM, 63(11), 2020. (pages 25 and 82.)

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. arXiv:1706.02677, 2017. (page 17.)

Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In Proc. ICCV, 2019. (pages 27, 28, 32, 35, and 60.)

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In Proc. NeurIPS, 2020. (pages 23, 37, 44, 58, and 60.)

Sophia Gu, Christopher Clark, and Aniruddha Kembhavi. I can't believe there's no images! Learning visual tasks using only language data. arXiv:2211.09778, 2022. (page 82.)

Samantha Guerriero, Barbara Caputo, and Thomas Mensink. DeepNCM: Deep nearest class mean classifiers. In Proc. ICLR-W, 2018. (pages 21 and 29.)

Xi Guo, Wei Wu, Dongliang Wang, Jing Su, Haisheng Su, Weihao Gan, Jian Huang, and Qin Yang. Learning video representations of human motion from synthetic data. In Proc. CVPR, 2022. (page 24.)

Yunhui Guo, Noel Codella, Leonid Karlinsky, John Smith, Tajana Rosing, and Rogerio Feris. A new benchmark for evaluation of cross-domain few-shot learning. In Proc. ECCV, 2020. (pages 28, 32, and 35.)

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In Proc. CVPR, 2006. (page 23.)

Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In Proc. ICCV, 2017. (page 35.)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proc. CVPR, 2016. (pages 16, 37, 42, 45, 53, 62, 67, 79, and 88.)

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In Proc. CVPR, 2020. (pages 8, 19, 22, 24, 27, 29, 32, 35, 37, 44, 58, 60, 64, 70, and 100.)

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proc. CVPR, 2022. (pages 23 and 24.)

Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? In Proc. ICLR, 2023. (page 81.)

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. JSTAE-ORS, 2019. (pages 66 and 92.)

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). arXiv:1606.08415, 2016. (page 63.)

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In Proc. ICCV, 2021a. (pages xvii, xviii, 66, 74, 90, and 91.)

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In Proc. CVPR, 2021b. (pages xvii, xviii, 66, 74, 90, and 91.)

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Patwary, Mostofa Ali, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. arXiv:1712.00409, 2017. (page 24.)

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In Proc. NeurIPS-W, 2014. (pages 20 and 82.)

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786), 2006. (page 23.)

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In Proc. NeurIPS-W, 2021. (page 87.)

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Proc. NeurIPS, 2020. (pages 25, 82, and 83.)

Elad Hoffer, Itay Hubara, and Daniel Soudry. Fix your classifier: The marginal value of training the last weight layer. In Proc. ICLR, 2018. (page 20.)

Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In Proc. CVPR, 2020. (pages 24 and 60.)

Sara Hooker. Moving beyond "algorithmic bias is a data problem". Patterns, 2021. (page 97.)

Andrew G Howard. Some improvements on deep convolutional neural network based image classification. arXiv:1312.5402, 2013. (page 17.)

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In Proc. CVPR, 2018. (page 16.)

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In Proc. ECCV, 2016. (page 17.)

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In Proc. CVPR, 2017. (page 16.)

Minyoung Huh, Pulkit Agrawal, and Alexei Efros. What makes ImageNet good for transfer learning? arXiv:1608.08614, 2016. (pages 3, 5, 27, and 90.)

Ben Hutchinson, Andrew Smart, Alex Hanna, Emily Denton, Christina Greer, Oddur Kjartansson, Parker Barnes, and Margaret Mitchell. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In ACM FAccT, 2021. (page 97.)

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. OpenCLIP, 2021. (page 79.)

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proc. ICML, 2015. (pages 17 and 63.)

Ashraful Islam, Chun-Fu Richard Chen, Rameswar Panda, Leonid Karlinsky, Richard Radke, and Rogerio Feris. A broad study on the transferability of visual representations with contrastive learning. In Proc. ICCV, 2021. (page 27.)

Phillip Isola. When faking your data actually helps – Learning vision from GANs, NeRFs, and noise, 2022. BMVC Keynote talk. (pages 25 and 82.)

Alexey Grigorevich Ivakhnenko. Polynomial theory of complex systems. SMC, 1971. (page 16.)

Ali Jahanian, Lucy Chai, and Phillip Isola. On the "steerability" of generative adversarial networks. In Proc. ICLR, 2020. (pages 25 and 82.)

Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. In Proc. ICLR, 2022. (pages 25 and 82.)

Dinesh Jayaraman and Kristen Grauman. Zero-shot recognition with unreliable attributes. In Proc. NeurIPS, 2014. (page 35.)

Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In Proc. CVPR, 2010. (page 15.)

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In Proc. ICML, 2021. (page 79.)

Jay Jiang and David Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In Proc. ROCLING, 1997. (page 40.)

Frederic Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. In Proc. ICCV, 2005. (page 15.)

Ioannis Kakogeorgiou, Spyros Gidaris, Bill Psomas, Yannis Avrithis, Andrei Bursuc, Konstantinos Karantzalos, and Nikos Komodakis. What to hide from your students: Attention-guided masked image modeling. In Proc. ECCV, 2022. (page 23.)

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In Proc. NeurIPS, 2020. (pages 18 and 44.)

Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without natural images. IJCV, 2022. (pages 24, 81, and 94.)

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In Proc. NeurIPS, 2020. (pages xiv, 6, 22, 59, 60, 61, 71, and 72.)

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv:1412.6980, 2014. (page 17.)

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. arXiv:2304.02643, 2023. (page 103.)

Byungsoo Ko and Geonmo Gu. Large-scale bilingual language-image contrastive learning. arXiv:2203.14463, 2022. (page 60.)

Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In Proc. CVPR, 2019. (page 41.)

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (BiT): General visual representation learning. In Proc. ECCV, 2020. (page 27.)

Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. CompRess: Self-supervised learning by compressing representations. In Proc. NeurIPS, 2020. (page 44.)

Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Mean shift for self-supervised learning. In Proc. CVPR, 2021. (page 23.)

Simon Kornblith, Jonathon Shlens, and Quoc Le. Do better ImageNet models transfer better? In Proc. CVPR, 2019. (pages 20, 27, 33, 35, 60, 65, 66, and 92.)

Simon Kornblith, Ting Chen, Honglak Lee, and Mohammad Norouzi. Why do better loss functions lead to less transferable features? In Proc. NeurIPS, 2021. (pages xv, 6, 9, 19, 20, 21, 58, 60, 62, 63, 95, 96, and 100.)

Jonathan Krause, Jia Deng, Michael Stark, and Fei-Fei Li. Collecting a large-scale dataset of fine-grained cars. In Proc. CVPR-W, 2013. (pages 66 and 92.)

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual Genome: Connecting language and vision using crowdsourced dense image annotations. IJCV, 2017. (page 102.)

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In Proc. NeurIPS, 2012. (pages 3, 14, 16, 17, 79, and 83.)

Anders Krogh and John Hertz. A simple weight decay can improve generalization. In Proc. NeurIPS, 1991. (page 17.)

Ashish Jith Sreejith Kumar, Rachel S. Chong, Jonathan G. Crowston, Jacqueline Chua, Inna Bujor, Rahat Husain, Eranga N. Vithana, Michaël J. A. Girard, Daniel S. W. Ting, Ching-Yu Cheng, Tin Aung, Alina Popa-Cherecheanu, Leopold Schmetterer, and Damon Wong. Evaluation of Generative Adversarial Networks for High-Resolution Synthetic Image Generation of Circumpapillary Optical Coherence Tomography Images for Glaucoma. JAMA Ophthalmology, 140(10), 2022. (pages 24 and 81.)

Christoph Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In Proc. CVPR, 2009. (pages 4, 27, and 35.)

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. FractalNet: Ultra-deep neural networks without residuals. In Proc. ICLR, 2017. (page 26.)

Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Tensor feature hallucination for few-shot learning. In Proc. WACV, 2022. (page 82.)

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In Proc. CVPR, 2006. (page 15.)

Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. In Proc. NeurIPS, 1989. (page 16.)

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 1998. (page 16.)

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In Proc. AISTATS, 2015. (pages 19 and 61.)

Daiqing Li, Junlin Yang, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In Proc. CVPR, 2021a. (page 24.)

Daiqing Li, Huan Ling, Seung Wook Kim, Karsten Kreis, Sanja Fidler, and Antonio Torralba. BigDatasetGAN: Synthesizing ImageNet with pixel-wise annotations. In Proc. CVPR, 2022. (pages 24, 25, and 81.)

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In Proc. NeurIPS, 2018. (pages 26 and 35.)

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. In Proc. ICLR, 2021b. (page 23.)

Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. arXiv:1708.02862, 2017. (page 44.)

Dekang Lin. An information-theoretic definition of similarity. In Proc. ICML, 1998. (pages xii, xiii, 34, 38, 51, and 53.)

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and Lawrence Zitnick. Microsoft COCO: Common objects in context. In Proc. ECCV, 2014. (pages 8, 18, 27, 36, and 104.)

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In Proc. CVPR, 2017. (pages 19 and 61.)

Tony Lindeberg. Feature detection with automatic scale selection. IJCV, 30(2), 1998. (page 15.)

Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. MP, 45(1), 1989. (page 66.)

Shusen Liu, Bhavya Kailkhura, Donald Loveland, and Yong Han. Generative counterfactual introspection for explainable deep learning. In GlobalSIP, 2019. (pages 25 and 82.)

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In Proc. ICCV, 2021. (page 16.)

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In Proc. CVPR, 2022. (page 16.)

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In Proc. ICLR, 2017. (page 17.)

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In Proc. ICLR, 2019. (page 17.)

David G Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60, 2004. (pages 1, 14, and 15.)

Zhichao Lu, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. Neural architecture transfer. PAMI, 2021. (pages 43 and 45.)

Thomas Lucas. Deep generative models : over-generalisation and mode-dropping. PhD thesis, Université Grenoble Alpes, 2020. (page 24.)

Alexandra Sasha Luccioni and David Rolnick. Bugs in the data: How ImageNet misrepresents biodiversity. arXiv:2208.11695, 2022. (pages 97 and 123.)

Xinyin Ma, Xinchao Wang, Gongfan Fang, Yongliang Shen, and Weiming Lu. Prompting to distill: Boosting data-free knowledge distillation via reinforced prompt. In Proc. IJCAI, 2022. (page 82.)

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In Proc. ECCV, 2018. (pages 6, 20, 32, and 44.)

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In Proc. CVPR, 2015. (page 82.)

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. Amass: Archive of motion capture as surface shapes. In Proc. ICCV, 2019. (pages 24 and 81.)

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. arXiv:1306.5151, 2013. (pages 8, 66, and 92.)

Chengzhi Mao, Augustine Cha, Amogh Gupta, Hao Wang, Junfeng Yang, and Carl Vondrick. Generative interventions for causal learning. In Proc. CVPR, 2021. (pages 25 and 82.)

Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In Proc. ACM-ICM, 2010. (page 89.)

Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In Proc. CVPR, 2016. (page 24.)

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5, 1943. (page 16.)

Lingling Meng, Runqing Huang, and Junzhong Gu. A review of semantic similarity measures in WordNet. IJHIT, 6(1), 2013. (page 39.)

T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. PAMI, 35(11), 2013. (pages 21 and 29.)

Thomas Mensink. Learning image classification and retrieval models. PhD thesis, Université de Grenoble, 2012. (page 14.)

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In Proc. ECCV, 2012. (pages 59 and 64.)

Elad Mezuman and Yair Weiss. Learning about canonical views from internet image collections. In Proc. NeurIPS, 2012. (page 36.)

Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. IJCV, 60, 2004. (page 15.)

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Proc. ICLR-W, 2013a. (page 5.)

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In Proc. NeurIPS, 2013b. (pages 5, 36, 40, and 52.)

George A Miller. WordNet: A lexical database for English. Commun. ACM, 38(11), 1995. (pages xiii, xviii, 5, 33, 35, 36, 37, 51, 53, 84, 86, 90, 99, and 122.)

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In Proc. CVPR, 2020. (page 23.)

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Unsupervised learning using sequential verification for action recognition. In Proc. ECCV, 2016. (page 23.)

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In Proc. NeurIPS, 2013. (page 17.)

Behnam Neyshabur, Srinadh Bhojanapalli, David Mcallester, and Nati Srebro. Exploring generalization in deep learning. In Proc. NeurIPS, 2017. (pages 26 and 35.)

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv:2112.10741, 2021. (page 25.)

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In Proc. ICVGIP, 2008. (pages 66 and 92.)

Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In Proc. ECCV, 2006. (page 15.)

Deniz Oktay, Carl Vondrick, and Antonio Torralba. Counterfactual image networks, 2018. URL https://openreview.net/forum?id=SyYYPdg0-. (pages 25 and 82.)

Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV, 42(3), 2001. (page 15.)

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. arXiv:2304.07193, 2023. (page 4.)

Michal Pándy, Andrea Agostinelli, Jasper Uijlings, Vittorio Ferrari, and Thomas Mensink. Transferability estimation using Bhattacharyya class separability. In Proc. CVPR, 2022. (pages 28 and 60.)

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. Neural Networks, 113, 2019. (page 97.)

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In Proc. CVPR, 2012. (pages 66 and 92.)

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In Proc. NeurIPS. 2019. (pages 67, 88, 89, 91, and 92.)

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. JMLR, 12, 2011. (pages 66 and 92.)

William Peebles, Jun-Yan Zhu, Richard Zhang, Antonio Torralba, Alexei A Efros, and Eli Shechtman. GAN-supervised dense visual alignment. In Proc. CVPR, 2022. (pages 24 and 81.)

Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In Proc. CVPR, 2007. (pages 1 and 14.)

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the Fisher kernel for large-scale image classification. In Proc. ECCV, 2010. (page 15.)

James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In Proc. CVPR, 2008. (page 15.)

Albert Pumarola, Jordi Sanchez-Riera, Gary P. T. Choi, Alberto Sanfeliu, and Francesc Moreno-Noguer. 3dpeople: Modeling the geometry of dressed humans. In Proc. ICCV, 2019. (pages 24 and 81.)

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Proc. ICML, 2021a. (pages 4, 6, 20, 44, and 45.)

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In Proc. ICML, 2021b. (pages 79,

81, 83, 84, and 97.)

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In Proc. ICML, 2021. (pages 25, 79, and 82.)

Suman Ravuri and Oriol Vinyals. Classification accuracy score for conditional generative models. In Proc. NeurIPS, 2019. (pages 25 and 81.)

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In Proc. CVPR-W, 2014. (pages 1, 4, 18, 58, and 60.)

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Proc. ICML, 2019. (pages xvii, xviii, 66, 74, 90, and 91.)

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In Proc. IJCAI, 1995. (page 39.)

Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Proc. ECCV, 2016. (pages 24 and 81.)

Marcus Rohrbach, Michael Stark, György Szarvas, Iryna Gurevych, and Bernt Schiele. What helps where–and why? Semantic relatedness for knowledge transfer. In Proc. CVPR, 2010. (page 38.)

Marcus Rohrbach, Michael Stark, and Bernt Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In Proc. CVPR, 2011. (page 39.)

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proc. CVPR, 2022. (pages xiv, 10, 25, 29, 78, 79, 80, 82, 83, 88, 94, 101, and 104.)

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6), 1958. (page 16.)

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. Nature, 323(6088), 1986. (page 16.)

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. IJCV, 115(3), 2015. (pages xi, xii, xviii, 3, 16, 18, 26, 27, 29, 33, 34, 35, 37, 38, 40, 46, 58, 65, 77, 79, 81, 83, 90, 99, and 105.)

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. In Proc. NeurIPS, 2022. (pages xii and 25.)

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. arXiv:1602.07868, 2016. (page 17.)

Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust ImageNet models transfer better? In Proc. NeurIPS, 2020. (page 44.)

Hadi Salman, Saachi Jain, Andrew Ilyas, Logan Engstrom, Eric Wong, and Aleksander Madry. When does bias transfer in transfer learning? arXiv:2207.02842, 2022. (page 97.)

Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In Proc. CVPR, 2018. (pages 7, 24, and 81.)

Mert Bulent Sariyildiz and Ramazan Gokberk Cinbis. Gradient matching generative networks for zero-shot learning. In Proc. CVPR, 2019. (pages 82 and 103.)

Mert Bulent Sariyildiz, Ramazan Gokberk Cinbis, and Erman Ayday. Key protected classification for collaborative learning. PR, 2020a. (page 20.)

Mert Bulent Sariyildiz, Julien Perez, and Diane Larlus. Learning visual representations with caption annotations. In Proc. ECCV, 2020b. (page 20.)

Mert Bulent Sariyildiz, Yannis Kalantidis, Diane Larlus, and Karteek Alahari. Concept generalization in visual representation learning. In Proc. ICCV, 2021. (pages xviii, 9, 10, 31, 58, 60, 65, 66, 75, 92, and 93.)

Mert Bulent Sariyildiz, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic ImageNet clones. In Proc. CVPR, 2023a. (pages 10, 11, and 78.)

Mert Bulent Sariyildiz, Yannis Kalantidis, Karteek Alahari, and Diane Larlus. No reason for no supervision: Improved generalization in supervised models. In Proc. ICLR, 2023b. (pages 9, 10, and 57.)

Axel Sauer and Andreas Geiger. Counterfactual generative networks. In Proc. ICLR, 2021. (page 25.)

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al.

LAION-5B: An open large-scale dataset for training next generation image-text models. arXiv:2210.08402, 2022. (pages 10, 29, 79, 88, and 97.)

Zhiqiang Shen and Marios Savvides. MEAL V2: Boosting vanilla ResNet-50 to 80%+ top-1 accuracy on ImageNet without tricks. In Proc. NeurIPS-W, 2020. (pages 37 and 44.)

Zhiqiang Shen, Zechun Liu, Zhuang Liu, Marios Savvides, Trevor Darrell, and Eric Xing. Un-mix: Rethinking image mixtures for unsupervised visual representation learning. In Proc. AAAI, 2022. (page 18.)

Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier. Rethinking deep active learning: Using unlabeled data at model training. In Proc. ICPR, 2021. (page 20.)

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Proc. ICLR, 2015. (pages 16, 37, 42, and 45.)

Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In Proc. ICCV, 2003. (page 14.)

Linda Smith and Michael Gasser. The development of embodied cognition: Six lessons from babies. Artificial life, 11(1-2):13–29, 2005. (page 17.)

Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In Proc. ICLR, 2018. (page 17.)

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In Proc. NeurIPS, 2013. (pages 27 and 35.)

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? Investigating data replication in diffusion models. arXiv:2212.03860, 2022. (page 97.)

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. JMLR, 15(1), 2014. (pages 17, 26, 35, and 58.)

Ryan Steed and Aylin Caliskan. Image representations learned with unsupervised pretraining contain human-like biases. In FAccT, 2021. (page 97.)

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proc. CVPR, 2015. (pages 16, 17, and 79.)

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proc. CVPR, 2016. (pages 37, 42, 44, and 58.)

Sora Takashima, Ryo Hayamizu, Nakamasa Inoue, Hirokatsu Kataoka, and Rio Yokota. Visual atoms: Pre-training vision transformers with sinusoidal waves. arXiv:2303.01112, 2023. (pages 81 and 94.)

Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proc. ICML, 2019. (pages 16, 43, 44, and 45.)

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The new data in multimedia research. Commun. ACM, 59(2), 2016. (page 44.)

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In Proc. ECCV, 2020a. (pages 17, 23, and 88.)

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In Proc. NeurIPS, 2020b. (page 44.)

Antonio Torralba and Alexei Efros. Unbiased look at dataset bias. In Proc. CVPR, 2011. (pages 36 and 102.)

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In Proc. ICML, 2021. (pages 37, 43, 44, and 45.)

Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. In Proc. NeurIPS, 2020. (pages 27 and 60.)

Nontawat Tritrong, Pitchaporn Rewatbowornwong, and Supasorn Suwajanakorn. Repurposing GANs for one-shot semantic part segmentation. In Proc. CVPR, 2021. (page 24.)

Tinne Tuytelaars and Cordelia Schmid. Vector quantizing feature space with a regular lattice. In Proc. ICCV, 2007. (page 15.)

Joost Van De Weijer and Cordelia Schmid. Coloring local feature extraction. In Proc. ECCV, 2006. (page 15.)

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with PixelCNN decoders. In Proc. NeurIPS, 2016. (page 23.)

Jan C Van Gemert, Cor J Veenman, Arnold WM Smeulders, and Jan-Mark Geusebroek. Visual word ambiguity. PAMI, 32(7), 2009. (page 15.)

Jan C Van Gemert, Cees GM Snoek, Cor J Veenman, Arnold WM Smeulders, and Jan-Mark Geusebroek. Comparing compact codebooks for visual categorization. CVIU, 114(4), 2010. (page 15.)

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The iNaturalist species classification and detection dataset. In Proc. CVPR, 2018. (pages xvii, 66, 75, and 92.)

Vladimir Vapnik. The nature of statistical learning theory. Springer science & business media, 1999. (page 19.)

Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In Proc. CVPR, 2017. (pages 24 and 81.)

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Proc. NeurIPS, 2017. (page 16.)

Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In Proc. ICCV, 2009. (page 79.)

Shashanka Venkataramanan, Bill Psomas, Ewa Kijak, Laurent Amsaleg, Konstantinos Karantzalos, and Yannis Avrithis. It takes two to tango: Mixup for deep metric learning. arXiv:2106.04990, 2021. (page 17.)

Shashanka Venkataramanan, Ewa Kijak, Laurent Amsaleg, and Yannis Avrithis. Alignmixup: Improving representations by interpolating aligned features. In Proc. CVPR, 2022. (page 18.)

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In Proc. ICML, 2019. (pages 18, 37, and 44.)

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In Proc. NeurIPS, 2016. (pages 27, 32, and 35.)

Riccardo Volpi, Diane Larlus, and Gregory Rogez. Continual adaptation of visual representations via domain randomization and meta-learning. In Proc. CVPR, 2021. (page 89.)

Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. Hoggles: Visualizing object detection features. In Proc. ICCV, 2013. (page 82.)

Bram Wallace and Bharath Hariharan. Extending and analyzing self-supervised learning across domains. In Proc. ECCV, 2020. (page 35.)

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In Proc. NeurIPS, 2019. (pages xii, xvii, xviii, 26, 66, 74, 90, and 91.)

Xiao Wang, Haoqi Fan, Yuandong Tian, Daisuke Kihara, and Xinlei Chen. On the importance of asymmetry for siamese representation learning. In Proc. CVPR, 2022a. (pages xiv, 60, 71, and 72.)

Yizhou Wang, Shixiang Tang, Feng Zhu, Lei Bai, Rui Zhao, Donglian Qi, and Wanli Ouyang. Revisiting the transferability of supervised pretraining: an MLP perspective. In Proc. CVPR, 2022b. (pages xv, 6, 22, 58, 61, 63, 69, 95, and 96.)

Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In Proc. CVPR, 2022. (page 23.)

Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In Proc. CVPR, 2011. (page 82.)

Paul Werbos. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. PhD thesis, Harvard University, 1974. (page 16.)

Yo whan Kim, Samarth Mishra, SouYoung Jin, Rameswar Panda, Hilde Kuehne, Leonid Karlinsky, Venkatesh Saligrama, Kate Saenko, Aude Oliva, and Rogerio Feris. How transferable are video representations based on synthetic data? In NeurIPS Datasets and Benchmarks Track, 2022. (pages 24 and 81.)

Ross Wightman, Hugo Touvron, and Hervé Jégou. ResNet strikes back: An improved training procedure in timm. In Proc. NeurIPS-W, 2021. (pages xiv, xviii, 6, 17, 21, 58, 59, 62, 66, 71, 72, 91, 92, and 93.)

Simon Winder, Gang Hua, and Matthew Brown. Picking the best daisy. In Proc. CVPR, 2009. (page 15.)

Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. ACL, 1994. (page 39.)

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In Proc. CVPR, 2018. (page 6.)

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—A comprehensive evaluation of the good, the bad and the ugly. PAMI, 41(9), 2018a. (page 35.)

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. PAMI, 41(9), 2018b. (page 82.)

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In Proc. CVPR, 2010. (pages 8, 36, 66, and 92.)

Tete Xiao, Xiaolong Wang, Alexei A Efros, and Trevor Darrell. What should not be contrastive in contrastive learning. In Proc. ICLR, 2021. (page 17.)

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In Proc. CVPR, 2017. (page 16.)

Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel, and Marc Niethammer. Robust and generalizable visual representation learning via random convolutions. In Proc. ICLR, 2021. (page 87.)

Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. arXiv:1905.00546, 2019. (pages 37 and 44.)

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. Joint learning of the embedding of words and entities for named entity disambiguation. In Proc. CONLL, 2016. (page 52.)

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. In Proc. EMNLP, 2020. (pages xiii, 52, and 53.)

Kaiyu Yang, Klint Qinami, Li Fei-Fei, Jia Deng, and Olga Russakovsky. Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the ImageNet hierarchy. In Proc. FAT, 2020. (pages 38 and 105.)

Kaiyu Yang, Jacqueline Yau, Li Fei-Fei, Jia Deng, and Olga Russakovsky. A study of face obfuscation in ImageNet. arXiv:2103.06191, 2021. (pages 105 and 106.)

Yanchao Yang and Stefano Soatto. FDA: Fourier domain adaptation for semantic segmentation. In Proc. CVPR, 2020. (page 7.)

Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In Proc. CVPR, 2020. (page 83.)

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Proc. NeurIPS, 2014. (pages 18, 32, 33, 35, 36, and 60.)

Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. In Proc. NeurIPS, 2020. (pages xiii, xv, 69, 70, 95, and 96.)

Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token ViT: Training vision transformers from scratch on imagenet. arXiv:2101.11986, 2021a. (pages 43 and 45.)

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. arXiv:2111.11432, 2021b. (page 4.)

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization strategy to train strong classifiers with localizable features. In Proc. ICCV, 2019. (pages 9, 17, 35, 37, 44, 58, and 103.)

Sangdoo Yun, Seong Joon Oh, Byeongho Heo, Dongyoon Han, Junsuk Choe, and Sanghyuk Chun. Re-labeling ImageNet: From single to multi-labels, from global to localized labels. In Proc. CVPR, 2021. (pages 44 and 54.)

Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In Proc. CVPR, 2018. (pages 4, 27, and 35.)

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-supervised learning via redundancy reduction. In Proc. ICML, 2021. (pages 23, 24, 44, and 61.)

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In Proc. ECCV, 2014. (pages 14 and 16.)

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. arXiv:1910.04867, 2019a. (pages 60 and 66.)

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. arXiv:1910.04867, 2019b. (pages 27 and 35.)

Hongyi Zhang, Moustapha Cisse, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In Proc. ICLR, 2018. (pages 9, 17, 37, 44, and 103.)

Jianyu Zhang, David Lopez-Paz, and Léon Bottou. Rich feature construction for the optimization-generalization dilemma. In Proc. ICML, 2022. (page 58.)

Richard Zhang, Phillip Isola, and Alexei Efros. Colorful image colorization. In Proc. ECCV, 2016. (pages 6, 19, 23, and 61.)

Yangmuzi Zhang, Diane Larlus, and Florent Perronnin. What makes an image iconic? A fine-grained case study. arXiv:1408.4325, 2014. (page 102.)

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In Proc. ICLR, 2021a. (page 82.)

Nanxuan Zhao, Zhirong Wu, Rynson W. H. Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? In Proc. ICLR, 2021b. (pages 22, 33, and 60.)

Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In Proc. ECCV, 2020. (page 24.)

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. PAMI, 2017. (pages xviii, 10, 27, 36, 87, 90, 102, and 122.)

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. Proc. ICLR, 2022. (pages 6 and 61.)