On the Generalization of Learned Structured Representations

Dittadi, Andrea

Publication date:
2022

Document Version
Publisher's PDF, also known as Version of record

DTU Compute
Department of Applied Mathematics and Computer Science

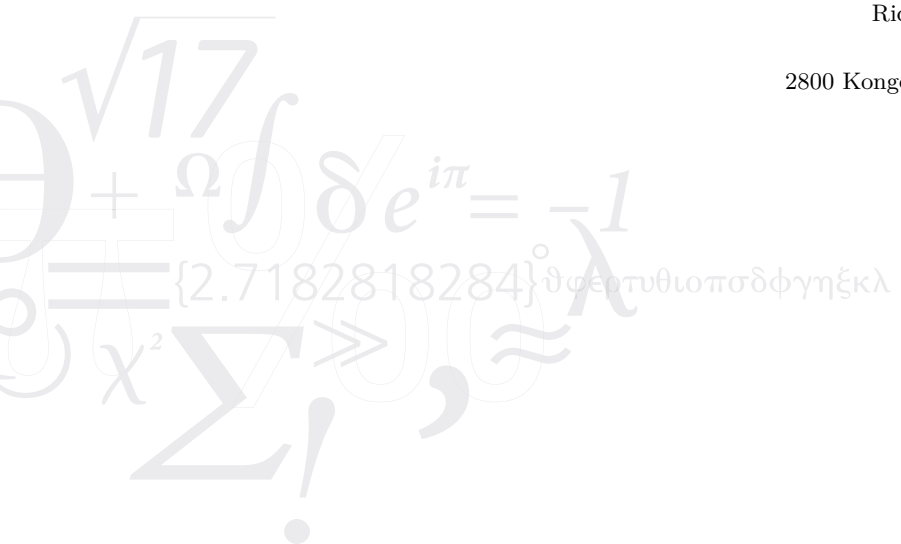
On the Generalization of Learned Structured Representations

Andrea Dittadi

Supervisor: Prof. Ole Winther

Co-supervisor: Prof. Thomas Bolander

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Richard Petersens Plads
Building 324
2800 Kongens Lyngby, Denmark



Summary

Despite tremendous progress over the past decade, deep learning methods generally fall short of human-level systematic generalization. It has been argued that explicitly capturing the underlying structure of data should allow connectionist systems to generalize in a more predictable and systematic manner. Indeed, evidence in humans suggests that interpreting the world in terms of symbol-like compositional entities may be crucial for intelligent behavior and high-level reasoning. Another common limitation of deep learning systems is that they require large amounts of training data, which can be expensive to obtain. In representation learning, large datasets are leveraged to learn generic data representations that may be useful for efficient learning of arbitrary downstream tasks.

This thesis is about structured representation learning. We study methods that learn, with little or no supervision, representations of unstructured data that capture its hidden structure. In the first part of the thesis, we focus on representations that disentangle the explanatory factors of variation of the data. We scale up disentangled representation learning to a novel robotic dataset, and perform a systematic large-scale study on the role of pretrained representations for out-of-distribution generalization in downstream robotic tasks. The second part of this thesis focuses on object-centric representations, which capture the compositional structure of the input in terms of symbol-like entities, such as objects in visual scenes. Object-centric learning methods learn to form meaningful entities from unstructured input, enabling symbolic information processing on a connectionist substrate. In this study, we train a selection of methods on several common datasets, and investigate their usefulness for downstream tasks and their ability to generalize out of distribution.

Resumé

På trods af enorme fremskridt i løbet af det seneste årti er *deep learning* generelt ikke i stand til at opnå systematisk generalisering på menneskeligt niveau. Det er en udbredt opffattelse, at eksplicit indfangning af den underliggende struktur i data skulle gøre det muligt for connectionistiske systemer at generalisere på en mere forudsigelig og systematisk måde. Faktisk tyder resultater fra eksperimenter med mennesker på, at det kan være afgørende for intelligent adfærd og ræsonnementer på højt niveau, at fortolke verden i form af symbollignende sammensatte enheder, der kan sammensættes og varieres. En anden almindelig begrænsning ved deep learning-systemer er, at de kræver store mængder træningsdata, som kan være dyrt at fremskaffe. I *representation learning* udnyttes store datasæt til at lære generiske datarepræsentationer, som kan være nyttige til effektiv indlæring af vilkårlige efterfølgende opgaver.

Denne afhandling omhandler indlæring af strukturerede repræsentationer. Vi undersøger metoder, der med lidt eller ingen supervision lærer repræsentationer af ustrukturerede data, som opfanger den skjulte, underliggende struktur. I den første del af afhandlingen fokuserer vi på repræsentationer, der udreder de forklarende faktorer for variation i dataene. Vi opskalerer indlæringen af udredte repræsentationer (*disentangled representations*) til et nyt robotdatasæt og gennemfører en systematisk, stor-skala undersøgelse af rollen, som forudindlærte repræsentationer spiller for generalisering uden for fordelingen til efterfølgende robotopgaver. Den anden del af denne afhandling fokuserer på objektcentrerede repræsentationer, som indfanger inputets kompositoriske struktur i form af symbollignende enheder, såsom objekter i visuelle scener. Objektcentrerede indlæringsmetoder lærer at uddrage meningsfulde enheder fra ustruktureret input, hvilket muliggør symbolsk informationsbehandling på et connectionistisk substrat. I denne undersøgelse træner vi et udvalg af metoder på flere ofte anvendte datasæt og undersøger deres anvendelighed til efterfølgende-opgaver og deres evne til at kunne generalisere uden for fordelingen.

Acknowledgements

First and foremost, I'd like to thank my supervisors, Ole Winther and Thomas Bolander. We started with an ambitious and rather vague plan that had something to do with combining symbolic AI and deep learning. Although I ended up being (mostly) a "deep learner" (sorry, Thomas!), I believe the spirit of this initial idea clearly shows in the topics I have been working on. Thank you, Ole, for your guidance as my main advisor, for always being unbelievably supportive and leaving me the freedom to explore, but always providing sharp feedback and being a reliable source of knowledge and wisdom. Thomas: although regrettably too few, I have learned a lot from our collaborations. I treasure our research discussions as well as our chats about life while having coffee at your fancy machine in building 322 or drinking beer in the nicest bars in Copenhagen.

I'd like to thank Bernhard Schölkopf and Stefan Bauer for warmly welcoming me to the Empirical Inference department at the MPI for Intelligent Systems: this was a turning point in my PhD and resulted in many successful collaborations, some of which are still ongoing. I'm also very grateful to the Causality group for many insightful discussions. I would like to especially thank Frederik Träuble for being a fantastic collaborator and friend since my first day at the MPI, and Stefan for being an invaluable mentor in the past couple of years and for always looking out for me.

I would also like to thank other people who have mentored me and helped me grow professionally, in particular Tom Cashman and Ben Lundell at Microsoft, and Peter Gehler and Francesco Locatello at Amazon. Special thanks to Francesco, who has been a great collaborator even outside of my internship at Amazon.

I'm also grateful for the many deep discussions and fun times at work and outside of work with my amazing PhD colleagues at DTU, including Valentin Liévin, Didrik Nielsen, Dimitris Kalatzis, Giorgio Giannone, Anders Christensen, and many more.

I'd like to thank all the collaborators in my projects: Ben Lundell, Bernhard Schölkopf, Darren Cosker, Felix Widmaier, Francesco Locatello, Frederik Drachmann, Frederik Träuble, Jamie Shotton, Koen Van Leemput, Manuel Wüthrich, Michele De Vita, Ole Winther, Olivier Bachem, Peter Gehler, Samuele Papa, Sebastian Dziadzio, Stefan Bauer, Stefano Cerri, Sveinn Pálsson, Thomas Bolander, Tom Cashman, Vaibhav Agrawal. I could not have done this without your invaluable help.

I would also like to thank everyone else I have co-authored a paper with: Anders Christensen, Anirudh Goyal, Darius Chira, Elliot Creager, Ilian Haralampiev, Lars Maaløe, Niki Kilbertus, Patrick Schwab, Simon Bing, Valentin Liévin, Yukun Chen. Thanks for involving me in your projects—I've learned a lot from all of you.

Finally, words cannot express how grateful I am to my friends and family—I wouldn't be here if it weren't for all of you. All my friends in Copenhagen, in Italy, and elsewhere, for always being there for me and for the many fun reunions around Europe. My family in Italy for their unconditional support. Anna, for being my pillar and enriching my life more than she knows.

Preface

This thesis was prepared at the Cognitive Systems section of the Department of Applied Mathematics and Computer Science (DTU Compute), Technical University of Denmark. It constitutes a partial fulfillment of the requirements for acquiring a PhD at the Technical University of Denmark.

The PhD project was supervised by Ole Winther and Thomas Bolander and it was financed by the Technical University of Denmark. The PhD project was carried out at the Technical University of Denmark in March 2018–April 2022, except for: a six-month stay at the Max Planck Institute for Intelligent Systems in Tübingen, Germany under the supervision of Bernhard Schölkopf and Stefan Bauer; a three-month internship at Microsoft Research in Cambridge, UK under the supervision of Tom Cashman and Ben Lundell; and a four-month internship at Amazon Web Services in Tübingen, Germany under the supervision of Peter Gehler and Francesco Locatello.

During the course of my PhD, I have worked on a variety of topics including representation learning, generative modeling with variational inference, disentanglement, object-centric learning, out-of-distribution generalization, and classical planning, with applications to robotics, video game playing, human pose modeling, medical imaging, image super-resolution, and medical time series. This work resulted in fifteen research papers (twelve peer-reviewed publications and three preprints); this dissertation is based on three of them.

Kongens Lyngby, 14 May 2022

Andrea Dittadi

Contents

List of Publications

This doctoral thesis is based on the following contributions:

1. **On the Transfer of Disentangled Representations in Realistic Settings**
   Andrea Dittadi,* Frederik Träuble,* Francesco Locatello, Manuel Wüthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, Bernhard Schölkopf
   ICLR 2021 (International Conference on Learning Representations)
   Dittadi et al., 2021b
   *Equal contribution.

2. **The Role of Pretrained Representations for the OOD Generalization of Reinforcement Learning Agents**
   Andrea Dittadi,* Frederik Träuble,* Manuel Wüthrich, Felix Widmaier, Peter Gehler, Ole Winther, Francesco Locatello, Olivier Bachem, Bernhard Schölkopf, Stefan Bauer
   ICLR 2022 (International Conference on Learning Representations)
   Dittadi et al., 2022b
   *Equal contribution. The order is random and can be swapped.

3. **Generalization and Robustness Implications in Object-Centric Learning**
   Andrea Dittadi, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, Francesco Locatello
   ICML 2022 (International Conference on Machine Learning)
   Dittadi et al., 2022a

The PhD student has also contributed to the following publications that are not included in this dissertation:

4. **Inductive Biases for Object-Centric Representations in the Presence of Complex Textures**
   Samuele Papa, Ole Winther, Andrea Dittadi
   Preprint: arXiv:2204.08479, 2022
   Papa, Winther, and Dittadi, 2022

5. **Image Super-Resolution With Deep Variational Autoencoders**
   Darius Chira,* Ilian Haralampiev,* Ole Winther, Andrea Dittadi,† Valentin Liévin†
   Preprint: arXiv:2203.09445, 2022
   Chira et al., 2022
   *Equal contribution, alphabetical order. †Equal advising, alphabetical order.

6. **Conditional Generation of Medical Time Series for Extrapolation to Underrepresented Populations**
   Simon Bing, Andrea Dittadi, Stefan Bauer,† Patrick Schwab†
   Preprint: arXiv:2201.08186, 2022
   Bing et al., 2022
   †Equal advising.

7. **Boxhead: A Dataset for Learning Hierarchical Representations**
   Yukun Chen, Frederik Träuble, Andrea Dittadi, Stefan Bauer, Bernhard Schölkopf
   NeurIPS 2021 Workshop on Shared Visual Representations in Human and Machine Intelligence
   Chen et al., 2021

8. **Full-Body Motion From a Single Head-Mounted Device: Generating SMPL Poses From Partial Observations**
   Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Tom Cashman, Jamie Shotton
   ICCV 2021 (IEEE/CVF International Conference on Computer Vision)
   Dittadi et al., 2021a

9. **Planning from Pixels in Atari with Learned Symbolic Representations**
   Andrea Dittadi,* Frederik Drachmann,* Thomas Bolander
   AAAI 2021 (AAAI Conference on Artificial Intelligence)
   Dittadi, Drachmann, and Bolander, 2021
   *Equal contribution.

10. **On disentangled representations learned from correlated data**
    Frederik Träuble, Elliot Creager, Niki Kilbertus, Francesco Locatello, Andrea Dittadi, Anirudh Goyal, Bernhard Schölkopf, Stefan Bauer
    ICML 2021 (International Conference on Machine Learning)
    Träuble et al., 2021

11. **Optimal Variance Control of the Score Function Gradient Estimator for Importance Weighted Bounds**
    Valentin Liévin, Andrea Dittadi, Anders Christensen, Ole Winther
    NeurIPS 2020 (Advances in Neural Information Processing Systems)
    Liévin et al., 2020

12. **Semi-supervised variational autoencoder for survival prediction**
    Sveinn Pálsson,* Stefano Cerri,* Andrea Dittadi,* Koen Van Leemput
    MICCAI 2019 Workshop on Brain Lesion
    Pálsson et al., 2019
    *Equal contribution.

13. **LAVAE: Disentangling Location and Appearance**
    Andrea Dittadi, Ole Winther
    NeurIPS 2019 Workshop on Perception as Generative Reasoning
    Dittadi and Winther, 2019

14. **Towards Hierarchical Discrete Variational Autoencoders**
    Valentin Liévin, Andrea Dittadi, Lars Maaløe, Ole Winther
    AABI 2019 (Symposium on Advances in Approximate Bayesian Inference)
    Liévin et al., 2019

15. **Learning to Plan from Raw Data in Grid-based Games**
    Andrea Dittadi, Thomas Bolander, Ole Winther
    GCAI 2018 (Global Conference on Artificial Intelligence)
    Dittadi, Bolander, and Winther, 2018

List of Figures

List of Tables

CHAPTER 1

Introduction

Despite the extraordinary progress made in the last decade in deep learning, human-level intelligence still seems out of reach. Major limitations of most contemporary methods include poor data efficiency and a lack of systematic generalization. *Representation learning* provides a possible way to alleviate the efficiency issue by learning to extract meaningful patterns in the data and represent them in a compact and reusable way, which should facilitate solving arbitrary downstream tasks, such as classification, abstract reasoning, planning, or robotic manipulation. In particular, representations that explicitly capture some of the *structure* in the data are believed to be beneficial for interpretability, efficiency of downstream learning, fairness, and human-level systematic generalization. In this dissertation, we will focus on areas of representation learning that are concerned with learning this structure: *disentangled* and *object-centric* representation learning.

Disentangled representations separately encode the ground-truth generative factors of variation of the data in a compact and reusable manner. Although recent years have seen several experimental studies on disentangled representation learning for image data, some questions remain unanswered. For example, (i) the generalization in downstream tasks is rarely investigated thoroughly, and (ii) these studies largely focus on toy datasets. One contribution of this thesis is a rigorous analysis of the generalization of disentangled representations in more realistic settings, including a large-scale study on the role of pretrained representations for the generalization in downstream robotic tasks. Chapter 3 provides an overview of these studies and a discussion of the key results. The original publications (Dittadi et al., 2021b; 2022b; referred to as Papers I and II in this dissertation) are included in Chapters 5 and 6.

Following an analogous line of reasoning, in a multi-object setting, representations should ideally capture the compositional structure of visual scenes in terms of objects. Separately representing objects and interpreting them as compositional building blocks should, in principle, enable complex symbolic reasoning, causal inference, and human-like systematic generalization. However, since *object-centric learning* developed relatively recently as a subfield of representation learning, there is still limited understanding of (i) the inductive biases for learning good object representations without supervision, (ii) their usefulness for downstream tasks, and (iii) their out-of-distribution generalization. In this dissertation, we present a systematic empirical study that thoroughly investigates how useful these representations are in practice, and how well common object-centric representation learning methods generalize out of distribution (points (ii) and (iii) above). Chapter 4 provides an overview of the study and a discussion of the major results. The original publication (Dittadi et al., 2022a; Paper III in this dissertation) is included in Chapter 7. In an additional paper not included in this thesis but briefly discussed in Section 4.2 (Papa, Winther, and Dittadi, 2022), we investigate architectural inductive biases that may help successfully separate objects with complex textures (point (i) above).

1.1 Thesis outline

The main body of this dissertation consists of three parts: In the first part (Chapter 2), we provide relevant background on the topics underlying the remainder of the thesis. In the second part (Chapters 3 and 4), we summarize our contributions, present the most salient results, and discuss them in depth. The third part (Chapters 5 to 7) contains Papers I, II, and III. We outline the structure of this thesis more in detail below.

Chapter 2 introduces the relevant background for this dissertation. After a brief overview of the history and motivation of representation learning, we discuss some desirable properties of learned representations. We focus in particular on their format, and introduce the concept of disentanglement. We then introduce *variational autoencoders*, a popular approach for generative modeling. We continue with a review of common approaches for disentangled representation learning based on variational autoencoders, and present the disentanglement metrics used in this thesis. Finally, we introduce unsupervised object-centric representation learning, and present a selection of popular methods that are relevant for this thesis.

Chapter 3 summarizes the main contributions of Papers I and II. These papers focus on learning (disentangled) representations in a robotic setting. In two large-scale studies, we investigate the relationship between properties of the representations, the performance on downstream tasks from simple factor prediction to challenging robotic tasks, and the generalization on these downstream tasks. We start by discussing our proposed dataset and the robotics context on which these publications are based. We then motivate and present the experimental studies, and discuss the key results and takeaways, leaving a more detailed analysis for Chapters 5 and 6.

Chapter 4 summarizes the main contributions of Paper III, where we perform an empirical study on unsupervised object-centric representation learning. In this study, we formulate three hypotheses based on common assumptions in the literature that are however typically left implicit, and systematically test them. More specifically, we investigate the usefulness of object representations for downstream models solving prediction tasks, and analyze their generalization under different types of distribution shifts at test time. Similarly to Chapter 3, in this chapter we discuss the key results and takeaways, leaving a more detailed analysis for Chapter 7.

Chapters 5, 6 and 7 contain Papers I, II, and III, respectively. Finally, we conclude in **Chapter 8** by summarizing and discussing the main contributions of this dissertation.

CHAPTER 2

Background

This chapter provides the relevant background for the contributions of this dissertation. We start in Section 2.1 by discussing the main motivations for representation learning, some key challenges in terms of generalization, and properties that are widely believed to be desirable in learned representations, such as disentanglement. In Section 2.2, we then provide an overview of variational autoencoders, which form the basis for some of the methods employed in this thesis, especially in the context of disentangled representation learning. We then continue in Section 2.3 with a review of disentangled representation learning, including common disentanglement metrics and a weakly supervised learning method that we will employ in Papers I and II (see Chapters 3, 5 and 6). Finally, in Section 2.4 we motivate and introduce object-centric representations and discuss a selection of recent methods for learning them without supervision (these will be used in Paper III; see Chapters 4 and 7).

2.1 Representation learning

2.1.1 The importance of data representation

The performance of machine learning algorithms is strongly affected by the way their input data is represented (Bengio, Courville, and Vincent, 2013; John, Kohavi, and Pfleger, 1994; Murphy, 2012; Ragavan et al., 1993). Representations that focus on relevant aspects of the data are easier to learn from, because the learning algorithm is spared the burden of having to infer which information is relevant for the task,

and isolate this information from irrelevant content. Moreover, such representations should be more robust to changes in the input that are irrelevant to the task at hand. These irrelevant changes are not necessarily noise in the classical signal processing sense (Oppenheim and Schafer, 2009; Rabiner and Gold, 1975); they can also be alterations in nuisance variables (or nuisance factors), i.e., proper factors of variation in the data that are coincidentally irrelevant for the task under consideration but may be relevant for others (Meehl, 1970; Fawzi and Frossard, 2016; Achille and Soatto, 2018, Section 2.2).

Traditionally, most machine learning applications involved linear models on top of hand-engineered features (Bengio, Courville, and Vincent, 2013; LeCun, Bengio, and Hinton, 2015; LeCun et al., 1998), whose purpose was to enrich or replace parts of the data in order to facilitate solving the task at hand. *Feature engineering* can be time-consuming as it typically relies on significant domain expertise and is often carried out manually by humans (LeCun, Bengio, and Hinton, 2015; Murphy, 2012, Section 1.2). For this reason, automated feature engineering has been the focus of a considerable amount of research in the past (Aha, 1991; Belongie, Malik, and Puzicha, 2001; Dalal and Triggs, 2005; Freeman and Roth, 1995; Freeman et al., 1996; Hirsh and Japkowicz, 1994; Lowe, 1999; Markovitch and Rosenstein, 2002; Matheus and Rendell, 1989; Scott and Matwin, 1999; Sutton and Matheus, 1991; Viola, Jones, and Snow, 2005), and is still relevant in some domains (Khurana, Samulowitz, and Turaga, 2018; Khurana et al., 2016; Lam et al., 2017; Nargesian et al., 2017; Zöller and Huber, 2021).

2.1.2 Deep neural networks

Deep learning approaches (LeCun, Bengio, and Hinton, 2015; Schmidhuber, 2015) do away with this expensive, often problem-specific feature engineering, and instead learn to extract data representations that are suitable for the task, together with the task itself. Such methods are based on *Deep Neural Networks* (*DNNs*), which consist of compositions of non-linear transformations, typically called *layers*. Each transformation computes a more abstract representation of the data using the less abstract, lower-level representations from the previous layers (Ballard, 1987). The deeper the layer—i.e., the further removed from the data—the more abstract and useful its representation of the data can be, as “more abstract concepts can be often

be constructed in terms of less abstract ones" (Bengio, Courville, and Vincent, 2013).[1] Furthermore, the complexity of the set of functions learnable by deep networks can grow exponentially with the network's depth (Eldan and Shamir, 2016; Montufar et al., 2014; Pascanu, Montufar, and Bengio, 2013; Raghu et al., 2017; Safran and Shamir, 2017; Telgarsky, 2016; but see also Håstad (1986), Håstad and Goldmann (1991), and Wegener (1987) for related results before the deep learning era) since the features at different layers can be *re-used and composed* in exponentially many ways (Bengio, Courville, and Vincent, 2013, Section 3.4).

Therefore, in addition to eliminating the need for the labor-intensive feature engineering on which traditional machine learning methods rely, DNNs can learn very flexible, abstract representations that are potentially even more suitable for a given task than problem-specific hand-engineered features—and they can do this directly from unstructured data. Their increased flexibility and ease of applicability have led DNNs to redefine the state of the art in a broad range of domains, with notable early successes in, e.g., image and video recognition (Ciresan et al., 2011; Krizhevsky, Sutskever, and Hinton, 2012; Sermanet et al., 2013; Simonyan and Zisserman, 2014; Zeiler and Fergus, 2014), natural language processing (Bordes, Chopra, and Weston, 2014; Collobert et al., 2011; Jean et al., 2014; Mikolov et al., 2011; Schwenk, Rousseau, and Attik, 2012; Seide, Li, and Yu, 2011; Socher et al., 2011; Sutskever, Vinyals, and Le, 2014), speech recognition (Hinton et al., 2012; Le et al., 2012; Mohamed, Dahl, and Hinton, 2011; Sainath et al., 2013), generation of images (Denton et al., 2015; Goodfellow et al., 2014; Mirza and Osindero, 2014) and raw audio (Oord et al., 2016), and reinforcement learning from unstructured visual input (Lillicrap et al., 2015; Mnih et al., 2015; Schulman et al., 2015a).

2.1.3 Learning generic representations for downstream tasks

Deep networks have also been employed to learn *generic* representations that are not tailored to a specific task and can be later used for arbitrary *downstream tasks* (Bengio, Courville, and Vincent, 2013). The learned representation function r, which maps a data point $\mathbf{x}$ to its representation $r(\mathbf{x})$, is then typically used as a black box for downstream learning (Brown et al., 2020a; Finn et al., 2016; Ha and Schmidhuber, 2018; Kingma et al., 2014; Peters et al., 2018; Stooke et al., 2021) or adapted (*fine-*

[1] This abstraction can be explicitly built into the architecture (e.g., pooling in convolutional neural networks), but it should also naturally occur when the task to be solved ultimately requires abstraction (e.g., classification of high-level concepts from raw unstructured data).

tuned) to the task at hand (Hinton and Salakhutdinov, 2006; Kolesnikov et al., 2020; Zbontar et al., 2021; Zhai et al., 2019). This approach is simply called *representation learning* (Bengio, Courville, and Vincent, 2013; Hamilton, 2020; Lesort et al., 2018; Rumelhart, Hinton, and Williams, 1986; Tschannen, Bachem, and Lucic, 2018; Wang and Isola, 2020), with the implication that no further objective is of interest at this stage, other than learning r itself.[2] These generic representation functions can be learned without supervision (Bengio et al., 2006; Chen et al., 2020a; Donahue and Simonyan, 2019; Hinton and Salakhutdinov, 2006; Radford et al., 2018; Ranzato et al., 2007; 2006), with self-supervision—i.e., technically unsupervised but trained with supervised learning using, e.g., auxiliary tasks (Ahmed et al., 2008; Doersch, Gupta, and Efros, 2015; Dosovitskiy et al., 2014; Gidaris, Singh, and Komodakis, 2018; Kolesnikov, Zhai, and Beyer, 2019) or contrastive methods (Chen et al., 2020b; He et al., 2020; Oord, Li, and Vinyals, 2018; Tian et al., 2020; Zbontar et al., 2021)—or with supervision on another dataset where numerous labeled examples are available (Girshick et al., 2014; Joulin et al., 2016; Kolesnikov et al., 2020; Sharif Razavian et al., 2014; Sun et al., 2017; Van Den Oord, Dieleman, and Schrauwen, 2014; Zeiler and Fergus, 2014).

A primary reason to do representation learning is that learning from raw data typically requires a large amount of labeled examples, which may be expensive or impossible to obtain. One way to alleviate this issue is to leverage other sources of data that at least partially share the structure of the target task. This paradigm is broadly referred to as *transfer learning* (Pan and Yang, 2009; Pratt and Jennings, 1996; Tan et al., 2018) as it involves transferring knowledge learned on one or more source tasks to a target task. The source task may or may not be supervised, and there may be a covariate distribution shift, a change in the task definition (e.g., unsupervised to supervised, or a different conditional distribution of the labels, or even a different label space), or both.

For example, Zhai et al. (2019) conduct a study of transfer learning in vision, where deep networks are trained on supervised, unsupervised, or self-supervised tasks on ImageNet (Deng et al., 2009), and the target tasks differ from the source tasks in terms of both the covariate distribution and the label space. Another example is domain adaptation (Pan et al., 2010; Wang and Deng, 2018), where the source and target

[2] In these cases, even if the representations are learned by training a DNN to solve a specific supervised (or self-supervised) task—such as classification on a massive dataset of real-world images—the ultimate goal is not to solve these (auxiliary) tasks as much as it is to learn meaningful representations for later use.

tasks are exactly the same (e.g., the conditional distribution of the classification labels $p(\mathbf{y}|\mathbf{x})$ is unchanged) but the covariates $\mathbf{x}$ undergo a distribution shift. It may also be the case that the source and target tasks differ while the data distribution $p(\mathbf{x})$ does not: for example, the source task could be unsupervised or self-supervised, and the target task may be classification or reinforcement learning (Grill et al., 2020; Laskin, Srinivas, and Abbeel, 2020). This can be seen as an instance of *semi-supervised learning* (Chapelle, Schölkopf, and Zien, 2006; Van Engelen and Hoos, 2020; Zhu, 2005).[3] Examples of this scenario include Papers I, II, and III (Dittadi et al., 2022a; 2021b; 2022b) as well as some of our works not included in this dissertation (Dittadi, Drachmann, and Bolander, 2021; Papa, Winther, and Dittadi, 2022; Träuble et al., 2021); however, in some of these works we also include experiments where the data distribution changes, e.g. with unseen values of the factors of variation (Dittadi et al., 2022a; 2021b; 2022b), synthetic image manipulations (Dittadi et al., 2022a), or from simulated data to the real world (Dittadi et al., 2021b; 2022b). Finally, an increasingly common approach in transfer learning is to train very large DNNs on massive generic datasets and adapt them to a wide range of downstream tasks (Brown et al., 2020b; Chen et al., 2020b; Devlin et al., 2019; Hénaff et al., 2021; Radford et al., 2021b; and see a discussion on the so-called *foundation models* in Bommasani et al. (2021)).

Even if data from the target task is available, learning from raw data may nevertheless be very inefficient when the task to be solved is particularly difficult, or when the function we are trying to learn is complex and highly-varying (Bengio, 2009, Section 2). This issue lies at the heart of the challenge of training deep architectures (Bengio, Courville, and Vincent, 2013, Section 10.1), which until the early 2010s was mostly based on layerwise unsupervised pretraining (Erhan et al., 2010). For example, even after recent significant significant progress in training very deep models, reinforcement learning algorithms are often trained from learned low-dimensional representations (Dittadi et al., 2022b; Finn et al., 2016; Stooke et al., 2021) or from the ground-truth state of the environment, if available (Ahmed et al., 2021; Lee, Hu, and Lim, 2021; Vinyals et al., 2019; Yu et al., 2020). Finally, even if learning end-to-end from raw data were practically feasible, it would still be advantageous (e.g., in terms of data and compute) to reuse a representation function that captures all the salient information in the data, rather than learn every new task from scratch.

[3] Roughly falling into the *unsupervised preprocessing* category of semi-supervised learning, more specifically *feature extraction* and *pretraining* (Van Engelen and Hoos, 2020, Section 5).

2.1.4 Desirable properties of learned representations

We introduced and motivated representation learning, and argued that *pretrained* representations can be beneficial for efficient learning of downstream tasks, in particular when obtaining target task data is costly. We will now consider desirable properties of the learned representation functions, more specifically in terms of their *information content* and their *structure*.

2.1.4.1 Information content

When learning a representation to be used for a downstream task, the representation should be *sufficient* for the task, i.e., it should retain all necessary information to solve it. On the other hand, we would like the representation to be *minimal*—i.e., to omit any information about the data that is not relevant for the task—and therefore invariant to irrelevant changes in the input (Bengio, Courville, and Vincent, 2013). Achille and Soatto (2018, Section 2.1) define a representation $\mathbf{z}$ of the data $\mathbf{x}$ to be sufficient for a task $\mathbf{y}$ (the target variable we wish to predict) if $I(\mathbf{z};\mathbf{y}) = I(\mathbf{x};\mathbf{y})$, i.e., the representation $\mathbf{z}$ contains as much information about the task as the data $\mathbf{x}$ itself (or, equivalently, $\mathbf{x}$ and $\mathbf{y}$ are conditionally independent given $\mathbf{z}$). Among the sufficient representations, the one that minimizes $I(\mathbf{x};\mathbf{z})$ is said to be minimal, because it contains just enough information about the data to allow solving the task, and is therefore invariant to the effect of nuisance factors (Achille and Soatto, 2018, Section 3).

The problem is that these notions are task-dependent: A sufficient representation for a task may not be sufficient for another task if the representation omits information that is irrelevant for the first task but necessary for the second. Even in a multi-task setting (Caruana, 1997; Ruder, 2017), a representation that is sufficient for *all* training tasks while minimizing $I(\mathbf{x};\mathbf{z})$ might be insufficient for a test task. This points to a fundamental trade-off between generality (usefulness for as many tasks as possible) and usefulness for specific tasks (potentially giving up usefulness for other tasks). Since future tasks are not necessarily known ahead of time, a sensible approach is to learn generic *task-agnostic* representations that make it possible to solve any downstream task that might be reasonably expected.[4]

[4]Different tasks "provide different views on the same underlying reality" (Bengio, 2009).

2.1.4.2 Structure

We now shift our focus from *what* information is in the representation to *how* this information is represented. To do so, we consider the generative aspect of the data. A common assumption is that the data is the result of a *generative process* where the *factors of variation* of the data tend to vary independently of each other, and typically few of them at a time vary in consecutive real-world inputs (Bengio, Courville, and Vincent, 2013; Schölkopf et al., 2021). These explanatory variables of the data, which we denote by the vector $\mathbf{g}$, are then mixed in a potentially highly complex and non-linear way according to a conditional distribution $p(\mathbf{x}|\mathbf{g})$, to give rise to the observed data $\mathbf{x}$. Ideally, given a data point $\mathbf{x}$, we would like to find the ground-truth factors $\mathbf{g}$ that generated it—or, following a Bayesian approach, the posterior distribution

$$p(\mathbf{g}|\mathbf{x}) = \frac{p(\mathbf{g})p(\mathbf{x}|\mathbf{g})}{\int_{\mathbf{g}} p(\mathbf{g})p(\mathbf{x}|\mathbf{g})\mathrm{d}\mathbf{g}} \tag{2.1}$$

of such factors. As we typically do not have access to the true generative model, nor to the ground-truth generative factors $\mathbf{g}$ underlying each observed data point $\mathbf{x}$, we learn representations $\mathbf{z} = r(\mathbf{x})$, with $\mathbf{x} \sim p(\mathbf{x}|\mathbf{g})$, that are in general different from $\mathbf{g}$.

Under the assumptions above, a good task-agnostic representation should preserve as much information as possible about the data, while separating, or *disentangling*, its explanatory factors of variation (Bengio, Courville, and Vincent, 2013, Section 3.5) in a compact and reusable manner (see Devereux et al. (2014) for evidence in humans). Although there is no generally agreed-upon definition of disentanglement, the core idea is that each specific factor z_j in a *disentangled representation* $\mathbf{z} = r(\mathbf{x})$ should only reflect the state of one ground-truth generative factor of variation g_i.[5] Intuitively, this should enable downstream processing systems (e.g., a classifier) to flexibly access the subset of factors in $\mathbf{g}$ that are relevant for the given task. In practice, disentangled representations have proven useful, e.g., for interpretability (Adel, Ghahramani, and Weller, 2018; Higgins et al., 2018), fairness of downstream models (Locatello et al., 2019a; Träuble et al., 2021), efficient downstream learning for reasoning tasks (Steenkiste et al., 2019), and generalization (Dittadi et al., 2021b; Locatello et al., 2020b).

[5] Sometimes, the definition of disentanglement is taken to be the converse, i.e., a change in one factor g_i should only affect one dimension of the representation (Chen et al., 2018). Arguably, this does not reflect the intuitive notion of disentanglement, since it allows multiple explanatory factors to be reflected in a single dimension of the representation. It has been termed *completeness* by Eastwood and Williams (2018), since a representation satisfying this definition would consist of elements that *completely* describe one (or more) factors of variation of the data.

Computer graphics provides a related perspective, where the data is generated by a known simulator given the generative factors $\mathbf{g}$, i.e., $p(\mathbf{x}|\mathbf{g})$ is available. These graphics codes present a compact description of a scene that aligns well with the desired representational properties discussed above. From this point of view, the task of inferring the generative factors of the data is sometimes also known as *inverse graphics* (Kulkarni et al., 2015), and has also been applied to inverting a *known* generative process such as a graphics engine (Eslami et al., 2016; Wu et al., 2017; Wu, Tenenbaum, and Kohli, 2017).

2.1.5 Challenges for generalization

Despite the benefits in terms of efficient learning of downstream tasks, especially when obtaining target task data is costly, there are practical issues when learning representations from raw data. First, there may be correlations or biases in the data that will be captured by the representations (Mehrabi et al., 2021; Torralba and Efros, 2011; Träuble et al., 2021), thereby limiting their generality and applicability.[6] Second, it might be desirable for learned representations to be invariant to some changes in the input, e.g., in terms of nuisance (task-irrelevant) variables in classification under domain shift (Anselmi et al., 2016; Pei et al., 2018) or in downstream reinforcement learning tasks (Zhang et al., 2020); if such invariances cannot be inferred from the training data distribution—e.g., if a nuisance variable such as lighting conditions in a scene is constant across the entire dataset—the learned representation function will not naturally exhibit them.

These issues can be problematic for generalization. For example, if a representation function is not invariant to lighting conditions, the representation $r(\mathbf{x})$ of an image $\mathbf{x}$ with unseen lighting conditions at test time may inaccurately represent other relevant information in the image. This issue is tackled, for example, in Papers I and II (Dittadi et al., 2021b; 2022b). Regarding correlations, a training dataset may for example contain images of a mountain landscape only in daylight, while city scenes are uniformly collected during day and night. Representation functions trained on such a dataset might not generalize well to photographs taken in the mountains at night, even though both mountain images and nighttime images are included in the

[6] This is also related to "shortcut learning" of surface statistics and spurious correlations. Relevant work in computer vision includes Hendrycks et al. (2021), Ilyas et al. (2019), Jo and Bengio (2017), and Xiao et al. (2021).

training data. In Träuble et al. (2021), we discuss training data correlations in the context of disentangled representation learning.

A conceptually simple solution is to collect more training data (Henighan et al., 2020; Kaplan et al., 2020; Sutton, 2019), assuming that in the limit of infinite data the spurious correlations and biases will disappear. However, despite the impressive generalization results of recent large-scale models trained on massive datasets (Brown et al., 2020a; Chowdhery et al., 2022; Radford et al., 2021a; Ramesh et al., 2022; 2021), their significance is sometimes called into question (Bender et al., 2021, Section 6.1; Heaven, 2021; Marcus, 2022), while bias (Abid, Farooqi, and Zou, 2021; Bender et al., 2021, Section 6.2) and training data memorization (Carlini et al., 2021) appear to persist.[7] An alternative solution is to attempt to directly resolve these biases during data collection and preprocessing, or in the learning algorithm itself (Mehrabi et al., 2021, Section 5.1).[8] Finally, we can mitigate some of these issues by introducing *inductive biases* (Mitchell, 1980), e.g., in the model architecture, in the training objective, or in the optimization procedure (Geirhos et al., 2020). Intuitively, among the many possible ways an algorithm could generalize, inductive biases should help choose one (an inductive bias is "any basis for choosing one generalization over another, other than strict consistency with the observed training instances" (Mitchell, 1980)). They typically derive from assumptions we make about the data or from potential constraints or requirements in the learning problem.

As discussed in Section 2.1.4, it has been argued that capturing and disentangling all factors of variation underlying the data—rather than attempting to build invariances in the representations—should lead to robust representations that generalize better (Bengio, Courville, and Vincent, 2013; Schölkopf et al., 2021). In Section 2.3, we will discuss the learning and evaluation of disentangled representations. Many popular disentanglement learning methods, including those in this dissertation, are based on variational autoencoders, which are covered next.

[7] It should be noted that the concept of generalization itself may be ill-posed as the training distributions become ever wider. What appear to be strong generalization capabilities might in fact be explained by (arguably very complex) interpolation. On the other hand, it becomes increasingly more difficult to define meaningful generalization tasks, since the training distributions are so wide.

[8] Note that there are other sources of bias and discrimination than the data itself (Mehrabi et al., 2021, Section 3.1).

2.2 Variational autoencoders

This section introduces variational autoencoders, a framework for variational inference in generative models that can be used for representation learning. In Section 2.3, we will then discuss common disentangled representation learning methods that are based on variational autoencoders.

Latent variable models. *Latent variable models* (LVMs) are probabilistic models with unobserved variables. In the following, we will denote the *observed variables* by $\mathbf{x}$, and the unobserved *latent variables* by $\mathbf{z}$. For simplicity, both $\mathbf{x}$ and $\mathbf{z}$ can be assumed to be vectors. The marginal distribution over the observed data is:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int_{\mathbf{z}} p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) \mathrm{d}\mathbf{z} \tag{2.2}$$

where $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ is the *joint distribution* over observed and latent variables, and $\boldsymbol{\theta}$ is a vector of model parameters. This probability distribution is typically called *marginal likelihood* or *model evidence*. Note that this is the marginal likelihood of a *single* data point. We are typically interested in the expectation of this quantity over the true data distribution, or more pragmatically, over the *empirical data distribution* $q(\mathbf{x})$:

$$\mathbb{E}_{q(\mathbf{x})}\left[p_{\boldsymbol{\theta}}(\mathbf{x})\right] = \frac{1}{N}\sum_{i=1}^{N} p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \int_{\mathbf{z}^{(i)}} p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) \mathrm{d}\mathbf{z}^{(i)} \tag{2.3}$$

where N is the size of the data set and the superscript denotes different data points. We will omit this outer expectation in the following, unless noted otherwise.

A simple and rather common structure for LVMs is:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z}) \tag{2.4}$$

where $p_{\boldsymbol{\theta}}(\mathbf{z})$ is the *prior distribution* over the latent variables and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is the conditional distribution of a data point given the latent variables, typically called *likelihood*. This structure suggests a generative interpretation of latent variable models: a data point $\mathbf{x}$ arises from a *generative process* (see Section 2.1.4.2) whereby latent variables $\mathbf{z}$ are first sampled from the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$ and then used for conditional sampling of an observation $\mathbf{x}$ according to the conditional distribution $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$.

Posterior inference. From the representation learning point of view (Section 2.1), it is typically of interest to invert this generative process and *infer* the posterior distribution of the latent variables given the observed data, i.e., intuitively, find the value

of the latent variables that gave rise to the observation (see Section 2.1.4.2, where in (2.1) we write the posterior of the latent variables of the *ground-truth* generative model). This is typically known as *posterior inference*.

The *posterior distribution* of the latent variables:

$$p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{x})} = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})}{\int_{\mathbf{z}} p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})\mathrm{d}\mathbf{z}} \tag{2.5}$$

is often intractable—e.g., when the likelihood is a non-linear function parameterized by a deep neural network—due to the lack of a practical estimator of, or an analytical solution to the integral in (2.2), which appears as denominator in (2.5).

Approximating the posterior with variational inference. The issue of the intractability of the latent posterior is addressed by approximating it with *sampling methods* or *variational inference*. Methods in the former class, such as Markov Chain Monte Carlo (MCMC), yield samples from the exact posterior distribution in the limit of infinite samples, so the approximation simply derives from having finite computational resources. The main limitation of these methods is that they do not scale well with dataset size and model complexity.

On the other hand, *variational inference* trades sampling for optimization, and provides a deterministic approximation to the posterior. In variational inference, we define a *variational distribution* $q_{\boldsymbol{\phi}}(\mathbf{z})$, with parameters $\boldsymbol{\phi}$, that approximates the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, and is therefore also known as *approximate posterior*. The variational distribution is then optimized to minimize a divergence from the true posterior, typically the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951):

$$D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}\right] \tag{2.6}$$

which is a non-negative quantity and is equal to 0 if and only if $q_{\boldsymbol{\phi}}(\mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ almost everywhere. Note that, in the standard case, $q_{\boldsymbol{\phi}}(\mathbf{z})$ approximates the posterior distribution for a *single* data point $\mathbf{x}$, and is derived or optimized separately for each example. In practice, the approximate posterior is restricted to a family of distributions that are flexible enough to yield a good approximation, but simple enough to allow for efficient optimization.

For any choice of $q_{\boldsymbol{\phi}}$, we have:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z})}\right] \geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z})}\right] = \mathcal{L}^{\mathrm{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) \tag{2.7}$$

using Jensen's inequality and the fact that the logarithm is a concave function. The quantity $\mathcal{L}^{\text{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x})$ is called *Evidence Lower BOund* (ELBO) as it is a lower bound to the marginal log-likelihood (or model evidence).

Crucially, maximizing this lower bound with respect to the variational parameters $\boldsymbol{\phi}$ is equivalent to minimizing the KL divergence between the true and approximate posterior distributions in (2.6). This can be shown by decomposing the log-likelihood as follows:

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(\mathbf{x}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x})\right] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x})}\frac{q_{\boldsymbol{\phi}}(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z})}\right] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z})}\right] + D_{\text{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{z}\,|\,\mathbf{x}))\ . \end{aligned} \tag{2.8}$$

Now, observe that the l.h.s. $\log p_{\boldsymbol{\theta}}(\mathbf{x})$ is fixed as we are only optimizing the approximate posterior. We can therefore conclude that maximizing the ELBO (2.7), which is the first term in (2.8), is equivalent to minimizing the KL divergence in the second term. Since the KL divergence is non-negative, this also provides an alternative derivation of the ELBO without using Jensen's inequality (cf. Eq. (2.7)).

Variational autoencoders. In contrast to traditional variational inference methods, where each data point has its own variational parameters that are optimized separately, *amortized variational inference* (Gershman and Goodman, 2014) uses function approximators like neural networks to share variational parameters across data points. Beside improving learning efficiency and allowing variational inference to scale to massive datasets, this amortization enables efficient inference on new data points at test time, whereas in traditional variational inference this would require an expensive optimization of the ELBO. In amortized variational inference, the approximate posterior of the latent variables is conditional on the observed data and denoted by $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$, so the ELBO for one data point can be written as follows:

$$\mathcal{L}^{\text{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\right] \leq \log p_{\boldsymbol{\theta}}(\mathbf{x})\ . \tag{2.9}$$

Variational Autoencoders (VAEs) (Kingma and Welling, 2014; Rezende, Mohamed, and Wierstra, 2014) are a framework for *amortized stochastic variational inference*, in which the expectation of the ELBO over the dataset

$$\mathbb{E}_{q(\mathbf{x})}\left[\mathcal{L}^{\text{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x})\right] \tag{2.10}$$

is maximized by jointly optimizing the inference model and the LVM (i.e., $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, respectively) with stochastic gradient ascent. The ELBO (2.9) to be maximized can be decomposed as follows (for one data point):

$$\begin{aligned}\mathcal{L}^{\text{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z})}\right] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\right] - D_{\text{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{z})) \end{aligned} \tag{2.11}$$

where the first term can be interpreted as negative expected *reconstruction error*, and the second term is the KL divergence from the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$ to the approximate posterior. In this setting, $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$ is often called *inference model* or *encoder*, while the likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})$ is called *decoder*. Typically, the prior is a fixed, isotropic Gaussian distribution with unit variance, such that the dependency on the parameters $\boldsymbol{\theta}$ can be dropped:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z};\, \mathbf{0}, \mathbf{I}) \ . \tag{2.12}$$

The approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$ is also a Gaussian with diagonal covariance matrix, but here the means and variances of each component are the output of an encoder network that takes the data as input:

$$q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) = \mathcal{N}(\mathbf{z};\, \mu_{\boldsymbol{\phi}}(\mathbf{x}), \text{diag}(\sigma_{\boldsymbol{\phi}}(\mathbf{x}))) \ . \tag{2.13}$$

VAE optimization. A crucial issue with the approach presented above is that the gradients of the ELBO cannot be naively backpropagated through the sampling step. However, for a rather large class of probability distributions (Kingma and Welling, 2014), a random variable can be expressed as a differentiable, deterministic transformation of an auxiliary variable with an independent marginal distribution. For example, if $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$ and $\mathbf{z} = \sigma_{\boldsymbol{\phi}}(\mathbf{x})\,\boldsymbol{\epsilon} + \mu_{\boldsymbol{\phi}}(\mathbf{x})$, then $\mathbf{z}$ is a sample from a Gaussian random variable with mean $\mu_{\boldsymbol{\phi}}(\mathbf{x})$ and standard deviation $\sigma_{\boldsymbol{\phi}}(\mathbf{x})$. Thanks to this *reparameterization*, $\mathbf{z}$ can be differentiated with respect to $\boldsymbol{\phi}$ by standard backpropagation. This widely used approach, called *pathwise gradient estimator*, tends to exhibit lower variance than the alternatives, which are typically based on the score function gradient estimator (Ranganath, Gerrish, and Blei, 2014; Williams, 1992).[9]

[9] However, the variance of this estimator can be reduced via *control variates* (Glasserman, 2004)—see, e.g., Mnih and Gregor (2014), Mnih and Rezende (2016), and Tucker et al. (2017). Additionally, there is a line of work that focuses on using multiple stochastic samples of the ELBO (Burda, Grosse, and Salakhutdinov, 2015, *importance-weighted ELBO*) to improve gradient estimates (Rainforth et al., 2018; Roeder, Wu, and Duvenaud, 2017); this includes our own work on optimal control variates for importance-weighted bounds (Liévin et al., 2020).

In the standard case, $\mathbf{z}$ follows a simple multivariate probability distribution such as a Gaussian with diagonal covariance. However, we can incorporate knowledge or assumptions about the generative process by defining a more structured probabilistic model. We apply this to object-centric generative modeling (see Section 2.4) in Dittadi and Winther (2019).

Rate–distortion tradeoff. From an information theory perspective, optimizing the variational lower bound (2.11) involves a tradeoff between rate and distortion, where the reconstruction term represents distortion and the divergence term represents rate (Alemi et al., 2018). A straightforward way to control the rate–distortion tradeoff is to use the β-VAE framework (Higgins et al., 2017a), in which the training objective (2.9) is modified by scaling the KL term with a scalar $\beta > 0$:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi},\beta}(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\right] - \beta D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x}) \,\|\, p(\mathbf{z})) \ . \tag{2.14}$$

In Section 2.3.1, we will review more fine-grained modulations of the KL term that have been proposed to encourage learning disentangled representations.

2.3 Disentanglement

In Section 2.1, we have discussed the importance of data representations and some desirable properties such as disentanglement. A substantial body of work spanning multiple decades has argued or demonstrated that learning disentangled representations is beneficial for a variety of purposes (e.g., Barlow, 1989; Bengio, Courville, and Vincent, 2013; Kulkarni et al., 2015; Lake et al., 2017; Locatello et al., 2019a; Peters, Janzing, and Schölkopf, 2017; Schmidhuber, 1992; Schölkopf et al., 2021; Steenkiste et al., 2019; Tschannen, Bachem, and Lucic, 2018). In this section, we provide an overview of common modern methods for learning disentangled representations, as well as popular metrics to assess their degree of disentanglement.

2.3.1 Learning disentangled representations

Most state-of-the-art methods for unsupervised disentanglement learning are based on the variational autoencoder (VAE) framework introduced in Section 2.2. The relative success of VAEs in disentanglement may be explained by the stochasticity of the

encoder, which promotes local orthogonality due to its diagonal covariance structure (Rolinek, Zietlow, and Martius, 2019). While approaches based on generative adversarial networks (GANs) exist, these typically only disentangle style and content and require some form of weak supervision (Chen et al., 2016; Lee et al., 2018; Mathieu et al., 2016). In the following, we present several VAE-based approaches for learning disentangled representations.

β-VAE. The β-VAE (see Eq. (2.14)) modulates the bottleneck capacity—i.e., the amount of information the model is allowed to represent in the latent variables—by modifying the KL divergence term of the ELBO (2.11) which intuitively acts as a regularizer for the approximate posterior. Increasing β encourages more structured (Burgess et al., 2018) but less informative representations, and corresponds to a lower rate and a higher distortion from the point of view of the rate–distortion tradeoff (Alemi et al., 2018). Decreasing β, on the other hand, leads to a higher rate and a lower distortion—i.e., more information in the latent variables and more accurate data reconstructions—but too weak a regularization may result in less structured and less useful representations (Bengio, Courville, and Vincent, 2013; Higgins et al., 2017a). Indeed, in our work we have observed that higher values of β tend to encourage disentanglement (Chen et al., 2021; Dittadi et al., 2021b; 2022b; Träuble et al., 2021), confirming previous results (Burgess et al., 2018; Higgins et al., 2017a). However, this behavior is not robust and it appears to strongly depend on the dataset (Locatello et al., 2019b, Fig. 15).

AnnealVAE. While this approach may be promising, a crucial issue is that the rate–distortion trade-off is, in some sense (and with some vigorous handwaving), independently present for each factor of variation. More concretely, let us assume there are two discrete factors of variation with the same number of possible values, and both affecting the color of an object. If the sizes of these objects differ significantly (e.g., one object constitutes the entire background while the other is a small cube in the foreground), the two factors will affect the reconstruction term $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ in the objective function to dramatically different extents. Given a bottleneck with limited capacity (i.e., a strong regularization in latent space, attainable for example by choosing $\beta \gg 1$) the factor with a minimal effect on the likelihood would be ignored. In general, the inference model will simply ignore the factors of variation that are not worth being modeled—in other words, those that require too high a rate compared to the modest reduction in distortion that modeling them would bring. We have observed this while running experiments for Dittadi et al. (2021b, not shown in the paper): some of the

β-VAEs trained with $\beta = 4$ did not model the rotation of the cube, and the learned generative model produced a cylinder (with the correct color and location), which can be interpreted as a cube averaged over all its possible rotations.

Noticing this issue, Burgess et al. (2018) proposed to address it by introducing the *AnnealVAE*, where the bottleneck capacity is progressively increased. The objective function then becomes:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi},\gamma}(\mathbf{x}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - \gamma\left|D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,\|\, p(\mathbf{z})) - C\right| \tag{2.15}$$

where $\gamma > 0$ plays a similar role to β, and C is annealed from 0 (which corresponds to a β-VAE with $\beta = \gamma$) to a potentially large positive value.

β-TCVAE and FactorVAE. Let us now consider the KL term of the ELBO (2.11), and decompose its expectation with respect to the data distribution $q(\mathbf{x})$ as follows:

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{x})}\left[D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,\|\, p(\mathbf{z}))\right] \\
&\quad= \mathbb{E}_{q(\mathbf{x})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\right] \\
&\quad= \mathbb{E}_{q(\mathbf{x})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\frac{q(\mathbf{x})q_{\boldsymbol{\phi}}(\mathbf{z})\prod_i q_{\boldsymbol{\phi}}(z_i)}{q(\mathbf{x})q_{\boldsymbol{\phi}}(\mathbf{z})\prod_i q_{\boldsymbol{\phi}}(z_i)}\right] \\
&\quad= \underbrace{D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})q(\mathbf{x}) \,\|\, q_{\boldsymbol{\phi}}(\mathbf{z})q(\mathbf{x}))}_{\text{or } \mathbb{E}_{q(\mathbf{x})}[D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \,|\, q_{\boldsymbol{\phi}}(\mathbf{z}))]} \\
&\qquad+ \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z})}{\prod_i q_{\boldsymbol{\phi}}(z_i)}\right] + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z})}\left[\log \frac{\prod_i q_{\boldsymbol{\phi}}(z_i)}{\prod_i p(z_i)}\right] \\
&\quad= I(\mathbf{x};\mathbf{z}) + D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}) \,\middle\|\, \prod\nolimits_i q_{\boldsymbol{\phi}}(z_i)\right) + \sum\nolimits_i D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(z_i) \,\|\, p(z_i)\right)
\end{aligned} \tag{2.16}$$

where z_i denotes the ith dimension of the latent variable $\mathbf{z}$.[10] This expression appears with a negative sign in the VAE objective function (i.e., the expected ELBO over the entire dataset (2.10)), and is therefore *minimized*. The first term is the *mutual information* between $\mathbf{x}$ and $\mathbf{z}$ with $\mathbf{x}, \mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})q(\mathbf{x})$. It is zero when $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = q_{\boldsymbol{\phi}}(\mathbf{z})$ almost everywhere, i.e., when the approximate posterior does not depend on the input. Penalizing this mutual information through the information bottleneck has been found to encourage compact and disentangled representations (Achille and Soatto, 2018; Burgess et al., 2018). On the other hand, it has also been argued that this term should not be penalized at all (Kim and Mnih, 2018; Makhzani et al., 2015;

[10] In fact, this decomposition holds for any partition of the dimensions of $\mathbf{z}$, but we focus on the case where each z_i is a scalar variable.

Zhao, Song, and Ermon, 2017). The second term is the *total correlation* (TC) of the variables $\{z_i\}$ under the distributions $\{q_{\boldsymbol{\phi}}(z_i)\}$. This is a generalization of the mutual information to multiple variables (Watanabe, 1960) and in this case penalizes *aggregate posteriors*

$$q_{\boldsymbol{\phi}}(\mathbf{z}) = \mathbb{E}_{q(\mathbf{x})}\left[q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})\right] = \frac{1}{N}\sum_{i=1}^{N} q_{\boldsymbol{\phi}}(\mathbf{z}^{(i)}\,|\,\mathbf{x}^{(i)}) \tag{2.17}$$

that do not factorize. Finally, the third term is the dimension-wise KL divergence from the prior to the aggregate posterior, which encourages each component of $q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})$ to be close to its prior (which is typically $\mathcal{N}(z_i;\,0,1)$).

Chen et al. (2018) argue that the total correlation term is what should be regularized in order to encourage disentanglement, and is the reason why β-VAEs tend to learn more disentangled representations when β is increased. Both the *β-TCVAE* (Chen et al., 2018) and the *FactorVAE* (Kim and Mnih, 2018) modify the standard ELBO objective by scaling the total correlation by a factor $\gamma > 1$, albeit using different estimators for this quantity. The objective function to be maximized can then be rewritten in terms of the original ELBO objective (2.11) as follows:

$$\begin{aligned}\mathbb{E}_{q(\mathbf{x})}\left[\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi},\gamma}(\mathbf{x})\right] &= \mathbb{E}_{q(\mathbf{x})}\mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}\,|\,\mathbf{z})\right] - I(\mathbf{x};\mathbf{z}) \\ &\quad - \gamma\, D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}) \,\middle\|\, \prod\nolimits_i q_{\boldsymbol{\phi}}(z_i)\right) - \sum\nolimits_i D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(z_i)\,\|\,p(z_i)\right) \\ &= \mathbb{E}_{q(\mathbf{x})}\left[\mathcal{L}^{\mathrm{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x})\right] - (\gamma-1)\, D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}) \,\middle\|\, \prod\nolimits_i q_{\boldsymbol{\phi}}(z_i)\right)\end{aligned} \tag{2.18}$$

where we explicitly include the expectation over the dataset, which is necessary for the decomposition in Eq. (2.16). The additional regularizer on the total correlation vanishes when $\gamma = 1$, resulting in the standard VAE objective.

DIP-VAE. Finally, Kumar, Sattigeri, and Balakrishnan (2018) claim that the standard VAE objective is not sufficient to encourage disentanglement, and propose to explicitly regularize a divergence between the prior and the aggregate posterior, to encourage the latter to be disentangled. The desired objective function (including the expectation over the data distribution) is then:

$$\mathbb{E}_{q(\mathbf{x})}\left[\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi},\gamma}(\mathbf{x})\right] = \mathbb{E}_{q(\mathbf{x})}\left[\mathcal{L}^{\mathrm{ELBO}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x})\right] - \gamma\, D\left(q_{\boldsymbol{\phi}}(\mathbf{z})\,\|\,p(\mathbf{z})\right) \tag{2.19}$$

where D is an arbitrary divergence. Note that, when D is the KL divergence, $D\left(q_{\boldsymbol{\phi}}(\mathbf{z})\,\|\,p(\mathbf{z})\right)$ is equal to the sum of the second and third terms in (2.16). Kumar, Sattigeri, and Balakrishnan (2018) introduce two ways of approximating the

additional divergence term, corresponding to two distinct optimization objectives: DIP-VAE-I and DIP-VAE-II.

2.3.2 Measuring disentanglement

Although a consensus has yet to be reached on a precise definition of disentanglement, various quantitative metrics have been proposed in an attempt to measure disentanglement based on the intuitive notions discussed earlier. Crucially, common disentanglement metrics do not always agree with each other in practice (Locatello et al., 2020c). While it is clear that some of them measure different notions than others, not all discrepancies are easily explained, especially in terms of the different results observed across datasets. Locatello et al. (2020a) present a thorough analysis and discussion, and provide recommendations for practitioners.

A first distinction to be made is whether a metric relies on *interventional data*—where we are allowed to perform interventions on the ground-truth factors of variation and assess how these affect the representations—or *observational data*—where we must estimate the statistical relationships between the learned representations and the ground-truth factors given a set of (annotated) examples.

In the interventional setting, two properties that are typically assessed are *consistency* and *restrictiveness* as defined by Shu et al. (2020), which measure the effect that intervening on a single factor of variation (or group of factors) has on the representation. In a *consistent* representation function, *fixing* a factor (or group of factors) and varying the others corresponds to fixing a subset of dimensions in the representation. In a *restrictive* representation function, *changing* a factor (or group of factors) with the others fixed corresponds to varying only a subset of dimensions in the representation.

In the observational setting, the metrics presented in this section typically focus on *disentanglement* and *completeness* in the sense of Eastwood and Williams (2018). Each dimension of a *disentangled* representation captures at most one factor of variation of the data: one factor may be encoded into multiple dimensions, but each of these dimensions must not encode any other factor. Conversely, in a *complete* representation, each factor of variation corresponds to at most one dimension in the representation: different factors may be mixed in one representation dimension, but no factor may be split over multiple dimensions.

The four properties outlined above are illustrated in Locatello et al. (2020a, Fig. 11). In the following, we introduce some of the most popular disentanglement metrics.

BetaVAE. The *BetaVAE* metric (Higgins et al., 2017a) is computed as the accuracy of a classifier that predicts which factor of variation has been fixed in a batch of image pairs. More specifically, for each batch, we choose a factor of variation i at random and sample a batch of pairs $(\mathbf{x}_1, \mathbf{x}_2)$ such that the ith factor has the same value in $\mathbf{x}_1$ and $\mathbf{x}_2$. All pairs in the batch have the same factor fixed, but the value can differ across pairs. For each image pair, we then compute the absolute value of the difference between the encoded representations of the two images, and finally average over the batch. The resulting vector:

$$\frac{1}{B}\sum_{n=1}^{B} |r(\mathbf{x}_1^{(n)}) - r(\mathbf{x}_2^{(n)})| \,, \tag{2.20}$$

where B is the batch size, is the input to the logistic regression model; the regression target is the index i of the factor that has been fixed in all the pairs in the batch. Each batch yields one data point for the downstream training of the regressor.

FactorVAE. Kim and Mnih (2018) discuss some weaknesses of the BetaVAE metric and propose to address them with the *FactorVAE* score, computed as follows: We first estimate the variance of each latent dimension and exclude unused dimensions (those with a small variance) from all subsequent computations. Then, we generate batches of samples where one randomly chosen factor is constant in each batch. For each generated batch, we estimate the representation dimension that encodes the fixed factor as the one that has the smallest variance (normalized by the global variance across the entire training set). The estimated dimension is one training sample for a majority vote classifier, and the (known) fixed factor is the corresponding target. The classifier rule is then defined as taking for each latent dimension the ground-truth factor that has the most votes from the training set.[11] The FactorVAE score is then the accuracy of this classifier on a held-out test set.

DCI. Eastwood and Williams (2018) argue that three distinct notions are relevant in this context: *disentanglement*, *completeness*, and *informativeness*. All three metrics are based on classifiers such as random forests or gradient boosted trees, one

[11] This can be achieved by repeating the step above for many batches, and constructing a matrix of size $K \times D$ (with K the number of factors and D the latent space dimensionality) where each entry denotes the number of times a given dimension was estimated to correspond to a specific factor. For each dimension d, the factor with most votes is the argmax of the dth column.

per factor of variation, each trained to predict the ground-truth factor value from the data representation $r(\mathbf{x})$. Informativeness is simply the average accuracy of the classifiers on a held-out test set. Disentanglement and completeness are based on the *importance matrix* obtained by concatenating the feature importances of each classifier. This matrix consists of the predictive importance of each representation dimension for each ground-truth factor. Disentanglement is maximized when each latent dimension has a positive importance for only one factor. Completeness measures the converse: it is maximized when only one latent dimension has a positive predictive importance for any given factor, i.e., each factor is only captured by one dimension in the representation. This framework is named *DCI* after the three metrics introduced. For simplicity, we will refer to the disentanglement metric defined by Eastwood and Williams (2018) as DCI.

MIG. Chen et al. (2018) propose to measure disentanglement with the Mutual Information Gap (MIG), based on the mutual information between each ground-truth factor of variation and each latent dimension. The mutual information gap for one ground-truth factor is defined as the difference between the highest and second-highest mutual information, normalized by the estimated entropy of that factor. When a factor is only represented by one latent dimension, this quantity is 1. The MIG score is then obtain by averaging over all ground-truth factors. Note that this metric in fact measures *completeness* (see p. 22).

Modularity and explicitness. Ridgeway and Mozer (2018) introduce *modularity* and *explicitness*. Modularity is computed in a similar fashion to the MIG, except that the gap is measured over factors of variation and for a given latent dimension, rather than the opposite. Modularity intuitively corresponds to disentanglement as defined in Section 2.1.4.2 and in Eastwood and Williams (2018). However, empirically, it appears to measure a different notion of disentanglement than other metrics (Locatello et al., 2020a, Section 6.1). Explicitness, on the other hand, measures how easily the factors of variation can be predicted from the representation (loosely corresponding to informativeness in Eastwood and Williams (2018)). It is computed for each factor of variation as the ROC-AUC (area under the receiver operating characteristic curve) of a logistic regression classifier trained to predict the ground-truth value of that factor. The global explicitness is the average of this quantity over all factors of variation.

SAP. The *Separated Attribute Predictability (SAP)* score proposed by Kumar, Sattigeri, and Balakrishnan (2018) is computed as follows: First, for each combination of factor of variation and latent dimension, we train a regression model or a classifier (depending on whether the factor is continuous or discrete) to predict the factor from the latent dimension. These result in a $K \times D$ *score matrix* containing the R^2 score (for continuous factors) or the accuracy (for discrete ones) for all combinations, computed on a test set. For each factor of variation, we then compute the difference between the highest and second-highest score. The SAP score is the average difference over all factors of variation. This difference will be maximal when each factor is only predictable from one dimension of the representation. Therefore, like the MIG, it measures *completeness* in the sense of Eastwood and Williams (2018) (see also p. 22).

IRS. The *Interventional Robustness Score (IRS)* proposed by Suter et al. (2019) performs interventions on the factors of variation and measures resulting changes in the representation. The *post-interventional disagreement* in a representation component z_k due to a generative factor g_j given a fixed value of g_i is defined as the distance (e.g., the ℓ_2-norm) between the expectation of z_k when we only fix g_i and when we also fix g_j (with $i \neq j$). Intuitively, we fix g_i and observe how robust z_k is when g_j changes. The IRS score then measures the (normalized) expected maximum disagreement over all factors of variation and their distributions, to assess the worst-case effect a change in nuisance factors (such as g_j) might have on the representation of g_i. Note that the interventional setting (see Pearl's do-calculus (Pearl, 2009)) is not necessary when there are no confounding correlations in the generative process, since in that case interventions are equivalent to regular conditioning. Suter et al. (2019, Section 5) also propose a method for estimating the IRS from purely observational data. Locatello et al. (2020a, Section 6.1) note that the IRS is not consistently correlated with the other disentanglement metrics.

2.3.3 Disentangling with limited supervision

Locatello et al. (2019b, Theorem 1) show that disentangled representation learning is impossible without supervision or appropriate inductive biases. They support this empirically in a large-scale experimental study, where they train the models presented in Section 2.3.1 on several synthetic datasets for disentanglement learning. In this study, they observe that hyperparameters and random seed appear to matter significantly more than the model type. Furthermore, since unsupervised model selection is

particularly challenging, it is necessary to directly evaluate disentanglement with the ad hoc metrics introduced in Section 2.3.2, which require ground-truth annotations of the factors of variation.

However, when some label information is available, it may be reasonable to use it instead as a direct supervision signal for learning disentangled representations: When a few observations are fully-labeled, this corresponds to the semi-supervised setting (Khemakhem et al., 2020; Locatello et al., 2020c; Sorrenson, Rother, and Köthe, 2020; but see also Klys, Snell, and Zemel (2018), Paige et al. (2017), and Reed et al. (2014) for previous related work). In Träuble et al. (2021, Section 5), we explore a different instantiation of semi-supervised learning (see Section 2.1.3) where representations learned in an unsupervised fashion are adapted (or aligned) to reflect new ground-truth information about the factors. On the other hand, when only weak labels are available (typically for the entire dataset), we can exploit them to learn disentangled representations by constructing a weakly supervised learning setting (Bouchacourt, Tomioka, and Nowozin, 2018; Hosoya, 2019; Locatello et al., 2020b; Shu et al., 2020), discussed below.

Weakly supervised disentanglement. We will briefly present here the weakly supervised approach proposed by Locatello et al. (2020b) that we will employ in Chapters 5 and 6 (Papers I and II). The key idea behind this method is that, while the ground-truth factors of variation are provably not identifiable in the i.i.d. case, they become identifiable given pairs of observations that differ in a subset of factors of size k. This subset of changing factors may differ from pair to pair, and even the number k of changing factors need not be fixed across all pairs. A possible justification for this setting is that changes in natural environments are caused by changes in a few factors of variation at a time (Földiák, 1991; Wiskott and Sejnowski, 2002), which is related to the *sparse mechanism shift* hypothesis (Schölkopf, 2019; Schölkopf et al., 2021). Unlike previous weakly supervised approaches that rely on group information (e.g., knowing which factors are changing between two observations), the method by Locatello et al. (2020b) only requires the number k of changing factors for each pair of observations to be known. In fact, they propose a practical algorithm that relaxes even this assumption by estimating k via a simple heuristic.

More concretely, the method computes approximate posterior distributions of the latent variables for a pair of images $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$, and then selects the k latent dimensions that *differ the most* in the posteriors of the two images, in terms of the KL

divergence $D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(z_i\,|\mathbf{x}^{(1)})\,\|\,q_{\boldsymbol{\phi}}(z_i\,|\mathbf{x}^{(2)}))$. We assume fully factorized approximate posteriors, and z_i denotes the ith dimension of $\mathbf{z}$. The k dimensions with a large KL divergence are considered to be changing between the two observations, while the others are considered to be unchanged. Since the KL divergence introduces an unnatural ordering in the input pair, in Papers I and II we modify the original definition by Locatello et al. (2020b) and use the *symmetrized* KL divergence (see Appendix B.1).

Then, given a symmetric averaging function a, the modified posterior distribution for the latent dimension j, and for $i \in \{1, 2\}$, is defined as follows:

$$\tilde{q}_{\boldsymbol{\phi}}^{(i)}(z_j\,|\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \begin{cases} q_{\boldsymbol{\phi}}(z_j\,|\mathbf{x}^{(i)}) & \text{if } z_j \text{ is inferred to be changing} \\ a(q_{\boldsymbol{\phi}}(z_j\,|\mathbf{x}^{(1)}), q_{\boldsymbol{\phi}}(z_j\,|\mathbf{x}^{(2)})) & \text{if } z_j \text{ is inferred to be shared} \end{cases}$$

such that the posteriors of the dimensions that are inferred to be unchanged between the two inputs are collapsed into the same distribution. The averaging function forces the approximate posterior of the shared latent variables to be the same for the two observations. Locatello et al. (2020b) propose to use the averaging functions from the *Multi-Level VAE* (ML-VAE) (Bouchacourt, Tomioka, and Nowozin, 2018) or from *Group-VAE* (GVAE) (Hosoya, 2019), and call the resulting methods *Ada-ML-VAE* and *Ada-GVAE*, respectively. In this dissertation, we will focus on Ada-GVAE (with the minor difference that we use a symmetrized KL divergence, as mentioned above), where the averaging function consists of averaging the means and variances of the (Gaussian) posteriors. The objective function for the pair of observations is a straightforward modification of the standard β-VAE objective:

$$\sum_{i\in\{1,2\}} \left(\mathbb{E}_{\tilde{q}_{\boldsymbol{\phi}}^{(i)}(\mathbf{z}\,|\mathbf{x}^{(1)},\mathbf{x}^{(2)})} \log(p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}\,|\mathbf{z})) - \beta D_{\mathrm{KL}}\Big(\tilde{q}_{\boldsymbol{\phi}}^{(i)}\big(\mathbf{z}\,|\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\big)\,\Big\|\,p(\mathbf{z})\Big)\right) \tag{2.21}$$

and it is optimized by drawing samples $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$ from a non-i.i.d. data distribution that accounts for the weak supervision assumptions discussed above.[12]

The authors empirically show that this approach (in both its variants) significantly improves the disentanglement of the learned representations, and that model selection can be successfully performed without explicit label supervision (i.e., the metrics from

[12] More formally, we can define the empirical joint distribution of each pair as follows:

$$q(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = q(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\,|k)q(k)\ , \tag{2.22}$$

where $q(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\,|k)$ selects a random pair of images that differ in k factors of variation and $q(k)$ is a distribution over k. In Chapters 5 and 6, we take $k = 1$ deterministically, or equivalently $q(k) = \delta(k - 1)$.

Section 2.3.2 are not needed). We remark, however, that in practice it seems to be necessary to show several pairs of observations with *only one* changing factor. This is evident from the original publication itself (Locatello et al., 2020b), where half of the training pairs are always differing only in one factor: when $k > 1$, the observations actually still have $k = 1$ with probability 0.5. We reached a similar conclusion while running experiments in Papers I and II (Dittadi et al., 2021b; 2022b), where $k = 1$ often yielded perfectly disentangled factors, while even $k = 2$ (without forcing $k = 1$ for half of the pairs) resulted in a dramatic performance drop.

2.4 Object-centric representations

2.4.1 Motivation

In Section 2.3, we introduced disentanglement and discussed how disentangled representations should be beneficial for downstream learning and generalization. The underlying assumption is that the data comes from a structured generative process with a few underlying factors of variation, and we wish to invert such a process by *disentangling* these factors. Although there is currently no precise definition of disentanglement, the consensus is that distinct factors of variation in the data should be represented separately from each other.

In general, however, visual scenes may contain a variable number of objects, which makes it less straightforward to define a plausible generative model of the data within the framework introduced so far. For example, if we learn a disentangled representation of visual scenes with one object such as the robotic setup in Papers I and II (Chapters 5 and 6; see Fig. 3.1), what should the representation of a scene with two objects be? When a second object is placed into the arena, either the representation stops being disentangled, or new factors of variation have to be introduced in order to represent the new object. One might argue that, if the task were to learn a representation of scenes where no more than one object is ever observed, it would be unreasonable to expect a model to generalize to multiple objects. However, the issue persists even when training on data with a variable number of objects: since there can always be more objects than ever observed in the training distribution, a robust model must have learned a mechanism to cope with the compositional structure of data that has discrete object-like building blocks. It is unclear how this may

be possible unless the representation has a modular structure and its size adapts to the corresponding observation.[13]

A second motivation for learning object representations can be found in the successes of *symbolic artificial intelligence* (AI) methods. Symbolic AI revolves around the idea that the abstractions necessary for reasoning and intelligent behavior are best represented by symbols. Historically, this approach has been at the basis of many of artificial intelligence's early successes, e.g., in automated planning (Fikes and Nilsson, 1971; Ghallab, Nau, and Traverso, 2004), theorem proving (Gelernter, 1959; Newell and Simon, 1956), and knowledge-based systems (Buchanan, Sutherland, and Feigenbaum, 1969). Despite their advantages, such as theoretical guarantees, interpretability, and systematic generalization, symbolic AI methods still require symbolic inputs and often rely on expert domain knowledge: Symbols have to be defined and grounded in the real world—this is known as the *symbol grounding* problem (Harnad, 1990; Searle, 1980; Steels, 2008)—and knowledge, in terms of domain-specific facts and rules, must be entered by humans into the system using a formal language (e.g., STRIPS (Fikes and Nilsson, 1971) and PDDL (Drew, 1998) in automated planning, or description logics (Baader and Nutt, 2003)).[14] While this approach may be reasonable in some cases, it is not acceptable in more general contexts that heavily involve learning and low-level perception. Because of these limitations, symbolic AI research has fallen out of favor in the deep learning era. However, many of these approaches are used to this day as part of the standard computer science toolbox, and many argue that symbol manipulation capabilities are necessary for overcoming the challenges of modern machine learning in terms of systematic generalization (Battaglia et al., 2018; Greff, Steenkiste, and Schmidhuber, 2020; Lake et al., 2017; Marcus, 2018; Marcus, 2003; Pearl, 2018; Schölkopf et al., 2021).

[13] Since the human working memory has a limited capacity and cannot hold more than a few objects simultaneously (Cowan, 2001; Fukuda, Awh, and Vogel, 2010; Kibbe and Leslie, 2019; Miller, 1956; Oberauer, 2019), we could also assume that the number of objects to be represented is bounded. However, here we focus on the general problem of representation learning in the multi-object setting, without necessarily limiting ourselves to scenarios that are realistic for human learning.

[14] Although in some cases this knowledge can be learned, observations are typically still required to be symbolic. This is the case, e.g., in action model learning for automated planning (Cresswell, McCluskey, and West, 2013; Mourao et al., 2012; Pasula, Zettlemoyer, and Kaelbling, 2007; Walsh and Littman, 2008; Yang, Wu, and Jiang, 2007; Zhuo et al., 2010). Note that, in Dittadi, Bolander, and Winther (2018), we do in fact learn a simple neural transition model in Sokoban from non-symbolic observations and successfully use it in a tree search algorithm for planning. However, (1) this approach relies on assumptions on the transition function for the domain, and (2) having a non-symbolic transition model prevents us from using optimized off-the-shelf planners that can automatically derive powerful heuristics to significantly improve planning efficiency.

On a similar note, machine learning methods have also been shown to benefit from symbol-like observations and from explicitly incorporating structure in a connectionist model. Recent successes can be found, e.g., in reinforcement learning (Ahmed et al., 2021; Berner et al., 2019; Vinyals et al., 2019) or in physical reasoning (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020) where structured observations are available via the internal state of a simulator. This access to ground-truth structured data is often necessary to solve complex tasks that require high-level skills such as reasoning and planning. Looking forward, it will be crucial to relax the assumption that ground-truth information about the state of the world is available, and learn to extract it instead.

Furthermore, there is evidence in cognitive psychology and neuroscience that humans perceive the world in a structured way, in terms of objects and their interactions (Spelke, 1990; Téglás et al., 2011; Wagemans, 2015). In fact, learning and reasoning about objects has been shown to develop in humans at an early age (Baillargeon, Spelke, and Wasserman, 1985; Dehaene, 2020; Spelke and Kinzler, 2007). Objects constitute compositional building blocks for higher-level cognitive tasks, and naturally enable systematic generalization outside of prior experiences (Dehaene, 2020).

Taking once again inspiration from human cognition, it has also been proposed that artificial intelligence should take a hybrid, *neuro-symbolic* approach, where direct sensory information is integrated with complex abstractions that allow for reasoning and planning (Marcus, 2018, Section 5.2). In this context, symbol manipulation can be performed by purely symbolic methods (Asai and Fukunaga, 2018; Ayton and Asai, 2021; Dittadi, Drachmann, and Bolander, 2021; Mao et al., 2019; Yi et al., 2018) or within a connectionist framework (Battaglia et al., 2018; Evans and Grefenstette, 2018; Greff, Steenkiste, and Schmidhuber, 2020; Pollack, 1990; Smolensky, 1990).

In general, object-centric representations are likely to play a crucial role in artificial learning systems that are able to reason, plan, and generalize systematically beyond their experience. In the remainder of this thesis, we will focus on how these representations are obtained from perceptual data alone, without any supervision, and regardless of how higher-level cognitive functions that manipulate these representations may be implemented. Note, however, that these issues are more generally related to the binding problem in neural networks, i.e., the "inability of contemporary neural networks to effectively form, represent, and relate symbol-like entities" (Greff, Steenkiste, and Schmidhuber, 2020).

2.4.2 Object-centric learning

In *object-centric representation learning*, we assume the data comes from a structured generative process based on discrete entities, their relationships, and a set of properties defining such entities and relationships. While disentangled representation learning (Section 2.3) is concerned with learning a uniform collection of factors of variation underlying the data, here we shift our focus to the compositional structure of data in terms of building blocks that we call *objects*. In this dissertation, we will focus on the most natural and intuitive case in the image domain, where objects are indeed *concrete* objects that are *visually* perceived in the world. However, the term "object" could be interpreted more generally as, e.g., spoken words or utterances, remembered entities, or abstract concepts and categories (Greff, Steenkiste, and Schmidhuber, 2020, Section 2.3).

The goal of object-centric learning is to obtain data representations that combine the richness of neural representations with the compositionality of symbols: for example, new objects can be created from unseen feature combinations, and objects can be composed in novel ways without their features interfering with each other (this is related to the "superposition catastrophe" (Bowers et al., 2014; Von Der Malsburg, 1986)). Although, intuitively, object-centric learning is strictly related to disentanglement learning—or may even be considered a special case thereof (Schölkopf et al., 2021, Section 6)—the traditional disentanglement framework cannot be straightforwardly applied in this case, as it assumes flat representations with a fixed vector format that imposes an arbitrary ordering of the dimensions.

The typical setting in object-centric representation learning is to assume the generative factors are a *set* of latent vectors $\{\mathbf{z}_i\}_{i=1}^N$ where N is the number of objects and each $\mathbf{z}_i$ contains the representation of one object in the observation $\mathbf{x}$. This *separation* (Greff, Steenkiste, and Schmidhuber, 2020, Section 3.1.1) of object representations is crucial for compositionality at the scene level. The number of objects could even be treated as a (discrete) generative factor of variation that defines the number of factors of variation for a given data point—e.g., in Dittadi and Winther (2019), a discrete random variable controls the number of latent vectors (*slots*) representing objects in each observation. Note that this *slot-based* approach to object separation is only one of the possible ways to tackle the problem—see Greff, Steenkiste, and Schmidhuber (2020, Section 3.3) for a discussion of more sophisticated strategies.

Although this thesis focuses on the common *autoencoding* setting for static images, object representations can also be learned via contrastive (Kipf, Pol, and Welling, 2019) or adversarial (Chen, Artières, and Denoyer, 2019b; Steenkiste et al., 2020) methods, or as intermediate representations in a larger model that is trained on a supervised task (Locatello et al., 2020d). Additional inductive biases can also be introduced through data, e.g., exploiting temporal information by observing sequences or by interacting with an environment (Kabra et al., 2021; Kipf et al., 2021; Kipf, Pol, and Welling, 2019; Löwe et al., 2020).

In the slot-based case considered here, all object representations $\mathbf{z}_i$ have a *common representational format* achieved by weight sharing across slots (Greff, Steenkiste, and Schmidhuber, 2020, Section 3.1.2). The shared format allows generalization between objects when using the representations downstream. This should lead, e.g., to the generalization of relations between objects independently of the context, which is relevant for reasoning, planning, and other high-level tasks. Finally, following the arguments in Section 2.3, it may be desirable for each object's features to be disentangled. This, together with the common representational format, is hypothesized to be beneficial for generalization to unseen feature combinations (Greff, Steenkiste, and Schmidhuber, 2020, Section 3.1.3).

2.4.3 Slot-based methods relevant for this thesis

In this section, we provide a brief overview of slot-based models and a high-level description of the methods relevant for this thesis. This section is based on parts of Paper III (Chapter 7) and its supplementary material (Appendix C).

Slot-based methods for object-centric learning can be categorized according to their approach to object separation (Greff, Steenkiste, and Schmidhuber, 2020). In models that use *instance slots*, each slot is used to represent a different part of the input. This introduces a routing problem, because all slots are identical but they cannot all represent the same object, so a mechanism needs to be introduced to allow slots to communicate with each other. In models based on *sequential slots*, the representational slots are computed in a sequential fashion, which solves the routing problem and allows to dynamically change the number of slots, but introduces dependencies between slots. In models based on *spatial slots*, a spatial coordinate is associated with each slot, introducing a dependency between slot and spatial location. In this

work, we focus on four scene-mixture models as representative examples of approaches based on instance slots (Slot Attention), sequential slots (MONet and GENESIS), and spatial slots (SPACE). What follows is a high-level overview of these four models.

MONet. In MONet (Burgess et al., 2019), attention masks are computed by a recurrent segmentation network that takes as input the image and the current *scope*, which is the still unexplained portion of the image. For each slot, a variational autoencoder (the *component VAE*) encodes the full image and the current attention mask, and then decodes the latent representation to an image reconstruction and mask. The reconstructed images are combined using the *attention* masks (*not* the masks decoded by the component VAE) into the final reconstructed image. The reconstruction loss is the negative log-likelihood of a spatial Gaussian mixture model (GMM) with one component per slot, where each pixel is modeled independently. The overall training loss is a (weighted) sum of the reconstruction loss, the KL divergence of the component VAEs, and an additional mask reconstruction loss for the component VAEs.

GENESIS. Similarly to MONet, GENESIS (Engelcke et al., 2020) models each image as a spatial GMM. The spatial dependencies between components are modeled by an autoregressive prior distribution over the latent variables that encode the mixing probabilities. From the image, an encoder and a recurrent network are used to compute the latent variables that are then decoded into the mixing probabilities. The mixing probabilities are pixel-wise and can be seen as attention masks for the image. Each of these is concatenated with the original image and used as input to the component VAE, which finds latent representations and reconstructs each scene component. These are combined using the mixing probabilities to obtain the reconstruction of the image. While in MONet the attention masks are computed by a deterministic segmentation network, GENESIS defines an autoregressive prior on latent codes that are decoded into attention masks. GENESIS is a proper probabilistic generative model, and it is trained by maximizing a modification of the ELBO introduced by Rezende and Viola (2018), which adaptively trades off the log-likelihood and KL terms in the ELBO.

Slot Attention. As our focus is on the object discovery task, we use the Slot Attention autoencoder model proposed by Locatello et al. (2020d). The encoder consists of a CNN followed by the Slot Attention module, which maps the feature map to a set of slots through an iterative refinement process. At each iteration, dot-product attention is computed with the input vectors as keys and the current

slot vectors as queries. The attention weights are then normalized over the slots, introducing competition between the slots to explain the input. Each slot is then updated using a GRU that takes as inputs the current slot vectors and the normalized attention vectors. After the refinement steps, the slot vectors are decoded into the appearance and mask of each object, which are then combined to reconstruct the entire image. The model is optimized by minimizing the MSE reconstruction loss. While MONet and GENESIS use sequential slots to represent objects, Slot Attention employs instance slots.

SPACE. Spatially Parallel Attention and Component Extraction (SPACE; Lin et al., 2020b) combines the approaches of scene-mixture models and spatial attention models. The foreground objects are segregated using bounding boxes computed through a parallel spatial attention process. The parallelism allows for a larger number of bounding boxes to be processed compared to previous related approaches. The background elements are instead modeled by a mixture of components. The use of bounding boxes for the foreground objects could lead to under- or over-segmentation if the size of the bounding box is not tuned appropriately. An additional boundary loss tries to address the over-segmentation issue by penalizing splitting objects across bounding boxes.

2.4.4 Measuring object separation

In this section, we define the segmentation metrics which we use in this dissertation to measure object separation. This section is based on the supplementary material for Paper III (see Appendix C).

Adjusted Rand Index (ARI). The Adjusted Rand Index (ARI) (Hubert and Arabi, 1985) measures the similarity between two partitions of a set (or clusterings). Interpreting segmentation as clustering of pixels, the ARI can be used to measure the degree of similarity between two sets of segmentation masks. Segmentation accuracy is then assessed by comparing ground-truth and predicted masks. The expected value of the ARI on random clustering is 0, and the maximum value is 1 (identical clusterings up to label permutation). As in prior work (Burgess et al., 2019; Engelcke et al., 2020; Locatello et al., 2020d), we only consider the ground-truth masks of foreground objects when computing the ARI. Below, we define the Rand Index and the Adjusted Rand Index in more detail.

The Rand Index is a symmetric measure of the similarity between two partitions of a set (Hubert and Arabi, 1985; Rand, 1971; Wagner and Wagner, 2007). It is inspired by traditional classification metrics that compare the number of correctly and incorrectly classified elements. The Rand Index is defined as follows: Let S be a set of n elements, and let $A = \{A_1, \ldots, A_{n_A}\}$ and $B = \{B_1, \ldots, B_{n_B}\}$ be partitions of S. Furthermore, let us introduce the following quantities:

- m_{11}: number of pairs of elements that are in the same subset in both A and B,
- m_{00}: number of pairs of elements that are in different subsets in both A and B,
- m_{10}: number of pairs of elements that are in the same subset in A and in different subsets in B,
- m_{01}: number of pairs of elements that are in different subsets in A and in the same subset in B.

The Rand Index is then given by:

$$\mathrm{RI}(A, B) = \frac{m_{11} + m_{00}}{m_{11} + m_{00} + m_{10} + m_{01}} = \frac{2(m_{11} + m_{00})}{n(n-1)} \tag{2.23}$$

and quantifies the number of elements that have been correctly classified over the total number of elements.

The Rand Index ranges from 0 (no pair classified in the same way under A and B) to 1 (A and B are identical up to a permutation). However, the result is strongly dependent on the number of clusters and on the number of elements in each cluster. If we fix n_A, n_B, and the proportion of elements in each subset of the two partitions, then the Rand Index will increase as n increases, and even converge to 1 in some cases (Fowlkes and Mallows, 1983). The expected value of a random clustering also depends on the number of clusters and on the number of elements n.

The Adjusted Rand Index (ARI) (Hubert and Arabi, 1985) addresses this issue by normalizing the Rand Index such that, with a random clustering, the metric will be 0 in expectation. Given the same conditions as above, let $n_{i,j} = |A_i \cap B_j|$, $a_i = |A_i|$,

and $b_i = |B_i|$, with $i = 1, \ldots, n_A$ and $i = 1, \ldots, n_B$. The ARI is then defined as:

$$\mathrm{ARI}(A, B) = \frac{\sum_{i,j} \binom{n_{i,j}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}} \tag{2.24}$$

which is 0 in expectation for random clusterings, and 1 for perfectly matching partitions (up to a permutation). Note that the ARI can be negative.

Segmentation covering metrics. Segmentation Covering (SC) (Arbelaez et al., 2010) uses the intersection over union (IOU) between pairs of segmentation masks from the sets A and B. How the segmentation masks are matched depends on whether we are considering the covering of B by A (denoted by $A \to B$) or vice versa ($B \to A$). We use the slightly modified definition by Engelcke et al. (2020):

$$\mathrm{SC}(A \to B) = \frac{1}{\sum_{R_B \in B} |R_B|} \sum_{R_B \in B} |R_B| \max_{R_A \in A} \mathrm{IOU}(R_A, R_B) \ , \tag{2.25}$$

where $|R|$ denotes the number of pixels belonging to mask R, and the intersection over union is defined as:

$$\mathrm{IOU}(R_A, R_B) = \frac{|R_A \cap R_B|}{|R_A \cup R_B|} \ . \tag{2.26}$$

While standard (weighted) segmentation covering weights the IOU by the size of the ground truth mask, mean (or unweighted) segmentation covering (mSC) (Engelcke et al., 2020) gives the same importance to masks of different size:

$$\mathrm{mSC}(A \to B) = \frac{1}{|B|} \sum_{R_B \in B} \max_{R_A \in A} \mathrm{IOU}(R_A, R_B) \ , \tag{2.27}$$

where $|B|$ denotes the number of non-empty masks in B. Since a high SC score can still be attained when small objects are not segmented correctly, mSC is considered to be a more meaningful and robust metric across different datasets (Engelcke et al., 2020).

Note that neither SC nor mSC are symmetric: Following Engelcke et al. (2020), we consider A to be the predicted segmentation masks and B the ground-truth masks of the foreground objects. As observed by Engelcke et al. (2020), both SC and mSC penalize over-segmentation (segmenting one object into separate slots), unlike the ARI. Both SC and mSC take values in $[0, 1]$.

CHAPTER 3

Study on representation learning in a robotic setting

After having introduced the necessary background in Chapter 2, we briefly outline the main contributions of this thesis. Here we take a more focused approach than in the original papers, and discuss the key motivations, conclusions, and limitations of the studies.

In this chapter, we introduce the motivation and experimental setting of Papers I and II (Chapters 5 and 6), summarize a few main results, and discuss the key takeaways. The most relevant background for this chapter is in Sections 2.1 to 2.3. We take a similar approach in Chapter 4, which is concerned with the experimental study in Paper III (Chapter 7).

3.1 Introduction and study design

In Papers I and II (Chapters 5 and 6), we focus on representation learning for downstream tasks in the context of robotics. We scale disentangled representation learning to a robotic setting and analyze the link between properties of the learned representations and different flavors of generalization in downstream tasks, spanning from ground-truth factor prediction to robotic pushing.

The setting is a simulated robotic platform consisting of a bowl-shaped stage with a flat floor, a monochromatic cube with a range of possible colors, and a robotic

arm with three degrees of freedom (Fig. 3.1, left). We consider two tasks: reaching the cube, or pushing it to a given target location. The platform has a real-world counterpart which we use for sim-to-real generalization experiments.[1] Images from the real-world platform are shown in the right panel in Fig. 3.1.

3.1.1 Dataset

Our first contribution, introduced in Paper I (Chapter 5), is a new annotated dataset for learning and evaluating disentangled representations in the robotic setting introduced above. A main advantage of this dataset is that it has a *direct downstream application to real-world robotics*. This has the beneficial side effect that on this dataset it is more difficult to learn useful representations that accurately capture the underlying structure of the data. Consequently, our dataset also provides a valuable testbed to challenge state-of-the-art methods for disentangled representation learning.

There are several aspects to the difficulty of our dataset:

1. It has a higher resolution (128x128) than most other datasets commonly used in disentangled representation learning (Fidler, Dickinson, and Urtasun, 2012;

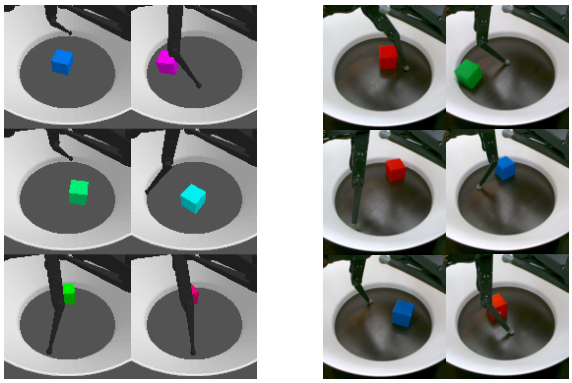


Figure 3.1. Random samples from the simulated (left) and real (right) dataset proposed in Paper I (Dittadi et al., 2021b).

[1] Our setting is derived from a more general setup with three robotic fingers that enables studying a wide range of robotic tasks from reaching to dexterous manipulation. This corresponds to Causal-World (Ahmed et al., 2021) in simulation and the TriFinger robot (Wüthrich et al., 2020) in the real world.

Gondal et al., 2019; Higgins et al., 2017a; Kim and Mnih, 2018; Reed et al., 2015), with the exception of SmallNORB (LeCun, Huang, and Bottou, 2004) that has the same resolution. Note that in this comparison we only consider datasets with precise annotations of the factors of variation: while there are datasets with labeled factors, such as CelebA (Liu et al., 2015), these have only qualitative annotations and, therefore, do not allow for quantitative evaluations or fine-grained control over the factors.

2. Our dataset has seven fully-annotated, fine-grained factors of variation; other datasets have fewer, except for MPI3D (Gondal et al., 2019) that also has seven.

3. Some of these factors of variation have correlations due to the interactions of the finger with the cube and the stage floor: the finger cannot be completely extended vertically and it cannot go through the cube. In another work not included in this dissertation (Träuble et al., 2021), although with simpler correlation structures, we show theoretically and empirically that disentangled representation learners might struggle when some factors of variation are correlated. However, we observe that the weakly supervised approach introduced in Section 2.3.3, which we also use in Papers I and II, may resolve this issue.

4. Some factors of variation have a significantly larger impact on the pixel-wise reconstruction loss. This could make it challenging to find the "sweet spot" for regularization strength in autoencoder-based disentanglement learners. In fact, we observe that the cube rotation—the factor with the smallest impact—is sometimes not captured by the representations when the regularization is too strong. See the related discussion in Section 2.3.1, in the paragraph that introduces AnnealVAE (p. 19).

5. Unlike most previous datasets, this dataset presents heavy occlusions. E.g., the tip of the finger may be hidden by the cube (or even exit the field of view, although this is not technically an occlusion), or the cube might be almost entirely hidden by the finger (see Fig. 3.1).

6. The factors of variations in the dataset are significantly more fine-grained than other datasets. This results in over 1.5 billion possible combinations of the seven factors, while the largest among the common disentanglement datasets only has one million combinations.

7. The fine granularity of the factors implies that our representations are trained on a rather small portion (less than 0.1%) of the space of possible combinations. By contrast, previous works on other datasets report training on the entire datasets, i.e., on all factor combinations.

A further advantage of our dataset is that it enables sim-to-real evaluation: in addition to one million annotated images of the simulated platform, it also includes over 1,800 annotated images of the real platform (Fig. 3.1, right).

3.1.2 Learning representations

To learn compact representations of simulated camera observations from our robotic platform, we choose the β-VAE for its simplicity. As discussed in Section 2.3.1, this is an extension to the variational autoencoder (VAE) that allows for the modulation of the information bottleneck capacity, thereby encouraging disentanglement. In addition to training β-VAEs in the standard unsupervised setting, we also use the Ada-GVAE approach proposed by Locatello et al. (2020b) to introduce weak supervision in the training procedure (see Section 2.3.3).

As mentioned above, this dataset is challenging for common disentanglement methods, which in our preliminary experiments failed to reconstruct the input images. For this reason, we increased the encoder and decoder depths and ran a hyperparameter sweep to determine the best configuration (including depth and width of the networks, presence and parameterization of residual connections, weight initialization of the networks, batch normalization, and dropout). The final architecture is over 4 times as deep as standard convolutional autoencoders for disentanglement learning (Locatello et al., 2020a), and it has over 20 times as many parameters (see Appendix A.1). Proposing a practical way to scale up disentanglement learning to more challenging settings constitutes our second contribution.

3.1.3 Study on generalization in downstream tasks

As a third contribution of Paper I (Chapter 5), we perform a reproducible large-scale study in which we train 1,080 variational autoencoders varying several hyperparameters, including supervision method (unsupervised or weakly supervised), bottleneck

capacity, and presence of noise in the input. The latter was an attempt—which turned out to be relatively successful—to learn encoders that would be more robust to strong distribution shifts such as sim-to-real. Further details on the hyperparameter search are presented in Section 5.3 and Appendix A.1.

In both Papers I and II, we *leverage the learned representation functions (the encoders) to learn downstream tasks, and investigate the relationship between representation properties and performance on downstream tasks, with a particular focus on out-of-distribution generalization.* The representation functions are pretrained and frozen, and a *downstream model* is trained to solve a specific task given the (learned) data representation as input. In Paper I, the task is to predict the ground-truth factors of variation; in Paper II, the task is either to reach the cube with the robotic finger, or to move it to a given location.

We formulate a framework for measuring generalization based on two scenarios:

- *OOD1*: The downstream model is evaluated out of distribution with respect to its training distribution, but *the representation functions are still in distribution.* This means that the representation $r(\mathbf{x})$ of an input $\mathbf{x}$ in the OOD1 set will be as good as representations of data in the downstream models' training distribution. Therefore, here we *purely test the generalization of downstream models* trained on representations with different structures and properties.

- *OOD2*: The downstream model is evaluated out of distribution with respect to both its own training distribution and that of the representation function. The key observation here is that, *since the representation functions are deep neural networks, they are prone to well-known generalization issues.* Therefore, for a data point $\mathbf{x}$ in the OOD2 set (when the encoders are out of distribution), the corresponding representation $r(\mathbf{x})$ *may not faithfully represent* $\mathbf{x}$.

Although often overlooked, this distinction is crucial when discussing generalization in representation learning, as it allows us to *separately study (1) the structure and properties of representations, and (2) the generalization of representation functions.*

For the sake of clarity, we now briefly introduce the concrete distribution shifts studied in Chapters 5 and 6 in the simulated setting.[2] We treat the cube color as a nuisance

[2]Although we also consider sim-to-real shifts, these cannot be precisely characterized in terms of factors of variations of the dataset. This could be solved, e.g., by introducing a binary factor of

factor (see Section 2.1.1) and consider distribution shifts affecting only this factor. In our setup, the cube color can take 12 possible values (with uniformly distributed hue in the HSV space, and maximum saturation and value). As shown in Fig. 3.2, we train the representation functions on 8 of these colors, chosen at random before running the study, and kept fixed. The held-out colors are used for OOD2 evaluation, i.e., with the representation function out of distribution. The downstream tasks—e.g., predicting the cube's position, or pushing the cube to a target location—are then trained on a *subset* of the colors that were used when training the representations (the 4 leftmost colors in the example in Fig. 3.2). When evaluating on the OOD1 colors, we gauge the generalization abilities of the downstream task, since in this case the representations should be accurate. When evaluating on the OOD2 colors, by contrast, we measure the generalization of the encoders as well. In Chapter 5, we consider three different splits of the VAE training colors. The "extrapolation" split shown in Fig. 3.2 (from downstream training colors to OOD1 colors) is the one we focus on in Chapter 6.

3.2 Key findings on disentanglement and factor prediction

In this section, we present and discuss the main results from Paper I on learning disentangled representations and solving downstream factor prediction tasks.

First, we demonstrate that the proposed architecture allows VAEs to reconstruct all relevant details of input images (see input–reconstruction pairs in Fig. 3.3(a)). This suggests that the latent representations in the trained models accurately capture all relevant information in the data (see Section 2.1.4.1). Arguably, this is necessary if

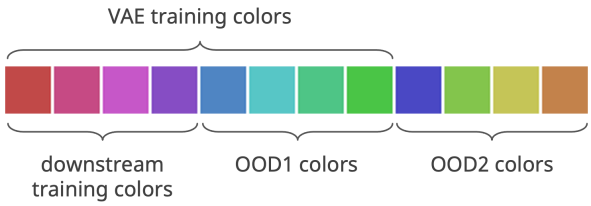
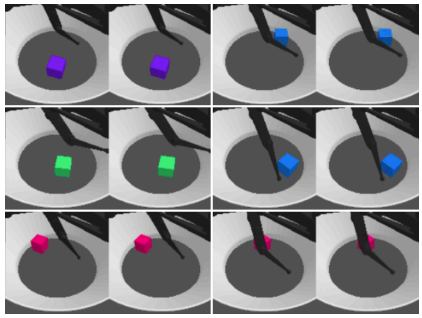
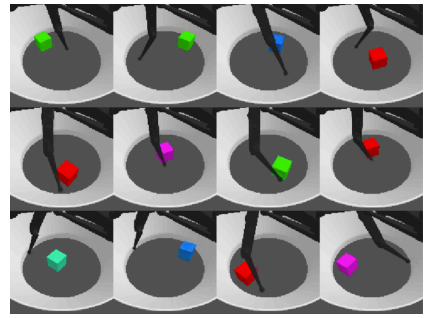


Figure 3.2. Illustration of the cube colors in the studies from Papers I and II.

variation that denotes simulation or real world, or by describing simulated and real settings via a set of complex factors of variation, such as surface characteristics and lighting conditions. This is, however, out of the scope of this dissertation and the research papers it is based on.



(a) Inputs and reconstructions.



(b) Random samples.

Figure 3.3. Input–reconstruction pairs and random samples of a VAE from our study.

we want to draw meaningful conclusions about the usefulness of representations with diverse structures (see Section 2.1.4.2): if relevant information is not retained by some representation functions, these will likely be less useful regardless of the structure of the information they do retain. In addition, although generative modeling is not a goal of this research effort, we remark that most of the learned models can also sample new high-quality images, as shown in Fig. 3.3(b).

Another key finding is that weak supervision as defined in Ada-GVAE (see Section 2.3.3) can successfully learn *fully disentangled* representations, as seen by visual inspection and from the remarkably high DCI Disentanglement scores (often above 0.99). The failure cases (when representations are not fully disentangled) typically occur when the latent space regularization is too strong and therefore one factor—the cube rotation, which contributes the least to the reconstruction error—is ignored (see discussion in Section 2.3.1). Conveniently, this allows for model selection in the weakly supervised case, since the entangled models are very likely to have worse unsupervised metrics (reconstruction loss and ELBO). Even in the cases where one factor of variation is ignored, those that are not ignored are often disentangled. In contrast, none of the unsupervised models learn disentangled representations, and almost all of them have lower DCI and MIG scores than any weakly supervised model (of the four disentanglement metrics considered in this study,[3] these are the one that we empirically observed, via visual inspection, to better correspond to our intuitive notion

[3]We can only compute *observational* metrics (DCI Disentanglement, MIG, Modularity, and SAP) on this dataset, as opposed to *interventional* metrics (see Section 2.3.2), because it does not provide interventional capabilities (it consists of a fixed set of images that is a rather small subset of all possible combinations of the factors of variation).

of disentanglement).

Thanks to this combination of unsupervised and weakly supervised approaches, we learn a collection of encoders that extract highly diverse representations. This allows us to draw sound conclusions from the observed relationships between relevant metrics. We will now focus in particular on disentanglement metrics and downstream factor prediction: How does the disentangling capability of the upstream representation function r affect the OOD performance of a downstream model that predicts the ground-truth factors of variation? A key insight from our findings—which are discussed below—is that the effect of disentanglement on generalization is not as straightforward as typically assumed.

First, as discussed above, it is crucial to define *in what sense* we are measuring generalization: If the representation function is OOD, the resulting representations will not be disentangled—in fact, they may not even be faithful to the data. For example, if the cube color in the input image never appeared in the VAE's training distribution, it is arguably unreasonable to expect the encoder to correctly interpret the unseen color. Indeed, in our experiments, encoders are typically unreliable out of distribution, as they sometimes fail at inferring even in-distribution factors. For example, when the color is OOD, the cube rotation is very often inferred incorrectly (see Fig. 3.4). This appears to hold regardless of the degree of disentanglement, which suggests that any disentanglement capabilities are lost when the data distribution shifts.

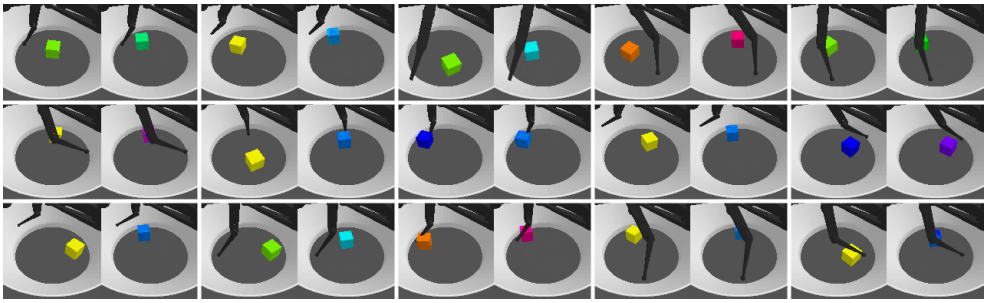


Figure 3.4. Each pair shows an OOD2 input (left) and reconstruction (right) by a VAE with perfectly disentangled factors of variation. The VAE was trained *without* input noise. Since the cube colors are out of distribution with respect to the encoder's training distribution, the mapping from an input image to a latent representation is not well defined. The encoder is especially likely to fail if the input color is not "similar enough" to a color in the training distribution (e.g., when the cube is yellow). Unsurprisingly, the generative model always reconstructs colors that were present in the training set.

Second, when the downstream model is an MLP, the degree of entanglement does not seem to matter at all. This is not surprising, since an MLP can be expected to disentangle factors of variation that are entangled in a representation. However, we observe a surprising phenomenon in the OOD1 case (i.e., when the encoder is *not* OOD): when the representation function *perfectly disentangles* the factors, the downstream models reliably attain a very low error on the task, while in other cases we observe a significant variance (see Fig. 5.4, left). On the other hand, disentanglement is not correlated with the OOD2 generalization of a downstream MLP, i.e., when the encoder is OOD (see Fig. 5.5).

In summary, in our experiments *disentanglement appears to matter only for OOD1 generalization, and what matters is mostly whether the representation is fully disentangled or not.* A possible explanation is that, when the prediction target is a (relatively simple) non-linear function of a single input feature, the optimization easily and reliably converges to the optimal solution where all other input features are ignored, including those that might be OOD at test time.

Another interesting observation is that adding random noise to the input of the representation function during training significantly improves its generalization, both in simulation (OOD cube colors) and to the real world. This is not entirely surprising, as noisy inputs have been shown theoretically and empirically to improve the robustness of neural networks. However, the extent of the improvement is undoubtedly significant, as seen qualitatively by comparing Fig. 3.4 and Fig. 3.5. Note that this falls into the OOD2 generalization category, and is particularly useful in practice in sim-to-real scenarios—Fig. 3.6 shows examples of the effectiveness of input noise for zero-shot sim-to-real generalization. Quantitative results are presented in Section 5.5 (Fig. 5.6).

3.3 Study on robotic tasks

In Paper II (Chapter 6), we investigate the role of pretrained representations on the performance and generalization of downstream reinforcement learning agents in the robotic setup introduced earlier, both in simulation and in the real world. To the best of our knowledge, this is the first systematic and extensive account of the OOD generalization of downstream reinforcement learning agents in robotics, and how this generalization is affected by characteristics of the upstream pretrained representation functions. In this section, we briefly outline the experimental setup for this study.

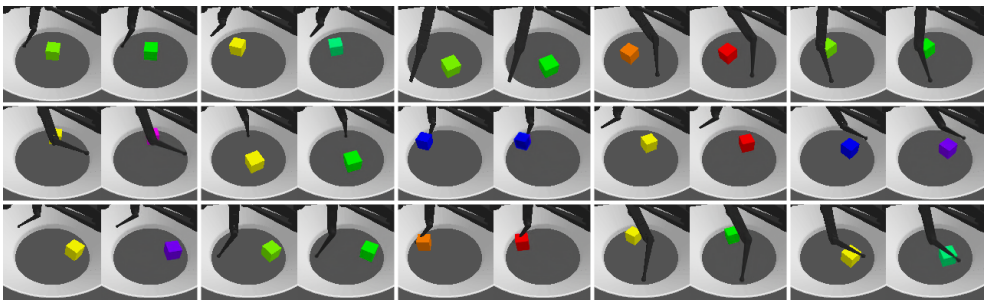


Figure 3.5. Each pair shows an OOD2 input (left) and reconstruction (right) by a VAE with perfectly disentangled factors of variation. The VAE was trained *with* input noise. Unsurprisingly, the generative model always reconstructs colors that were present in the training set. However, unlike in the noiseless case in Fig. 3.4, the other factors of variation are inferred relatively well.

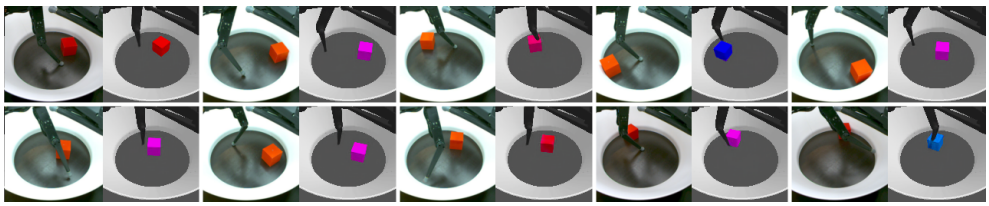
We then discuss a few key insights in Section 3.4, leaving a more extended exposition for Chapter 6.

In this study, we evaluate distribution shifts within a framework that is analogous to the one in Paper I. However, the experimental study in this robotic setting is significantly more complex and demanding:

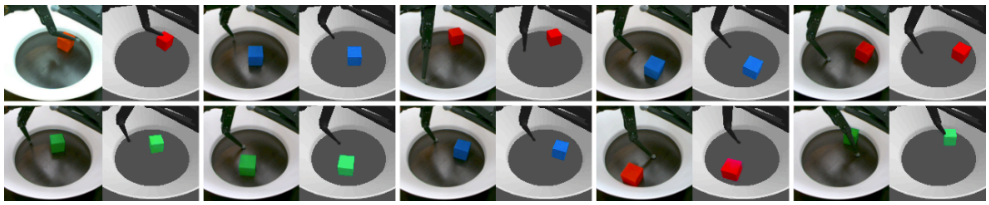
- The downstream task is considerably more challenging: While predicting one factor of variation involves training a downstream model on 10k samples for a few thousand steps, we train our reinforcement learning policies for 3M steps for pushing.

- While in the simple prediction task we can precompute the representations of the entire training and test sets, in the reaching and pushing tasks we must encode each input image at runtime. This is an additional source of computational cost, besides the mere fact that the task is harder.

- Since the objective function in reinforcement learning is typically based on some form of expected cumulative reward, its gradients with respect to the model parameters are not computable in closed form and have to be estimated (Mohamed et al., 2020). Gradients in reinforcement learning are thus generally harder to estimate than in supervised learning tasks such as factor prediction, where the training loss is directly differentiable with respect to all parameters. This, together with the fact that the tasks are harder in the first place, require us to



(a) Trained without noise.



(b) Trained with noise.

Figure 3.6. Each pair shows a real-world image input (left) and its reconstruction by trained VAEs (right). This is an OOD2 scenario. The model trained with input noise (b) infers the ground-truth factors of variation significantly more accurately than the model trained without noise (a).

use many (10 for pushing, 20 for reaching) random seeds for downstream RL training. This further increases the already high overall computation time by one order of magnitude.

Thus, we opt to reduce the computational burden by limiting the number of pretrained representations, and use only a subset of the models trained for Paper I.

We train reinforcement learning policies with SAC (Haarnoja et al., 2018b) to either reach the cube in the arena or push it to a given target location. For reaching, we measure success by the fractional progress made by the tip of the robot finger from its initial position to the surface of the cube. For pushing, we measure the fractional volumetric overlap between the cube and the goal (which is defined as a cube of the same size).

We select a subset of the representation functions from Paper I and train 11,520 downstream policies in total. For each encoder, we use 20 random seeds for training downstream policies on reaching, and 10 seeds on pushing. Since disentanglement is one of the central themes of this study, we also explore the effect of L1 regulariza-

tion on the input, in an attempt to encourage the downstream policies to disregard nuisance factors and therefore be more robust to distribution shifts.

The input to the downstream policies is the concatenation of: (1) the input representation $r(\mathbf{x})$, where r is the frozen, pretrained representation function, (2) the ground-truth angles and velocities of the robot joints, and (3) the target position and orientation of the cube, in the pushing task. The cube's position and orientation are thus the only pieces of information that must be contained in $r(\mathbf{x})$, since all other relevant quantities are available in the ground-truth state observation. Despite this, solving these tasks proved to be difficult—in fact, they can be challenging even from complete ground-truth information (Ahmed et al., 2021).

3.4 Key findings on robotic tasks

One of the motivations for introducing the annotated robotics dataset in Paper I was to scale up disentangled representation learning to more realistic settings and eventually evaluate its usefulness on more relevant downstream tasks than factor prediction on toy datasets, such as reinforcement learning on a robot. In Paper II, we deliver on this promise. However, since the results on disentanglement are mostly negative, we expand our study to investigate more in general how the generalization of downstream policies relates to a variety of metrics that can be computed on the representations before training the policies.

We start by analyzing the effect of disentanglement on the generalization of the trained policies. In Paper I, we observed that disentanglement is not beneficial in the OOD2 scenario, but it can be helpful for OOD1 generalization (i.e., when the encoder is in distribution) only when the representations are perfectly disentangled. In Paper II, on the other hand, we find the role of disentanglement to be negligible: it appears not to be beneficial even when the encoder is kept in distribution and even if it perfectly disentangles the factors of variation.

As in Paper I, we test the encoder's robustness (OOD2 generalization) by evaluating the policies on unseen cube colors in simulation, as well as on the real robot. Here we also test on an unseen shape (a sphere) both in simulation and in the real world. In simulation, where it is feasible to evaluate a large number of policies on hundreds of episodes each, we observe that training the representation functions with input noise

significantly improves OOD2 generalization, which is in line with the conclusion we reach on the simpler prediction setting of Paper I. In particular, some of the policies trained in simulation generalize surprisingly well zero-shot to the real robot, without any fine-tuning or domain randomization during training. Crucially, the best policies in the real-world setting tend to be the best ones in the OOD2 setting in simulation, for example with unseen cube colors. Therefore, *we can use a policy's OOD2 performance in simulation to predict how it will perform on the real robot.*

Perhaps the most valuable takeaway from this study, however, is that we can interpret the out-of-distribution performance on the simple downstream tasks from Paper I as *generalization scores*, to be added to the collection of representation metrics. Thus, if we are interested in a specific type of generalization for downstream policies—e.g., corresponding to a specific training distribution and test-time distribution shift—we can simply replicate a similar train/test scenario on a simple prediction task and expect the OOD performance on such a task to be predictive of the policies' OOD performance. We can use the toy task as a *proxy task* to predict the performance of downstream models on a *target task*. Notably, the distribution shift need not be exactly the same in the proxy and target tasks: for example, the results on substantially different OOD2 shifts—OOD cube colors in simulation, OOD shape in simulation, and sim-to-real shift—appear to be correlated.

Let us consider a concrete example to highlight the relevance of these results. Assume we have trained a large number of representation functions to be used as vision backbones for downstream reinforcement learning. We train downstream policies in simulation and deploy them on a real robot. It is reasonable to assume that we should train many policies for each representation function, varying hyperparameters and random seed. Fortunately, our study suggest that we can leverage simple proxy tasks to drastically reduce the number of policies that need to be trained: First, we train factor prediction models for all representation functions and evaluate them on images from the real robot. Then, we select the encoders with the best OOD performance and run a hyperparameter sweep for the policies using only this subset of encoders as upstream models. Note that, since the OOD2 performance of the policies in simulation and in the real world are also correlated, we could then further reduce the cost of deploying multiple policies to the real robot by filtering out the ones that do not perform particularly well on the simulated OOD2 setting. However, the most impactful advantage is arguably the possibility to pre-select representations that are more likely to lead to robust downstream policies.

Finally, although not discussed in the paper, it would not be surprising if these results held beyond reinforcement learning in robotics. In fact, we could hypothesize that the correlations between the performances on proxy and target tasks may primarily depend on how similar the downstream models and tasks are.

Regarding the *similarity of downstream models*, in Paper II we indeed observe significant correlations only when the downstream model for factor prediction is an MLP, like the neural networks in the RL agents. Note that this is probably a soft constraint: First, each RL agent actually consists of multiple MLPs (the policy, value, and Q networks), therefore a direct comparison is not straightforward. Second, in Paper I we observe that the behaviors of MLPs with up to 3 hidden layers are similar to each other, but different from gradient boosted trees, random forests, and k-nearest neighbors.

The *similarity of the proxy and target downstream tasks* can probably be expressed in terms of the mutual information between the prediction targets of the two tasks. Assume that the data is defined by a set of ground-truth (generative) factors $\mathcal{G}$, and that solving the proxy task T_p and the target task T_t requires information about two subsets $\mathcal{G}_p, \mathcal{G}_t \subset \mathcal{G}$ of factors.[4] We might then observe that models for T_p that are downstream of a representation function r tend to exhibit some properties, e.g., good in-distribution performance or robustness to a specific kind of distribution shift. Intuitively, if $\mathcal{G}_p$ and $\mathcal{G}_t$ are "similar", we may expect the same properties to hold to some extent for models that are trained downstream of r to solve task T_t.[5] Note that, if the tasks are entirely unrelated (i.e., $\mathcal{G}_p \cap \mathcal{G}_t = \varnothing$), some properties may still be consistent: for example, if r is robust to distribution shifts, downstream models trained on T_p and T_t might both exhibit good OOD2 performance; and if r is *not* robust, both downstream models will probably not perform well in an OOD2 setting. Although these relationships are not straightforward to characterize, we found some empirical evidence that partially supports this hypothesis: The OOD1 performance of the policies is particularly correlated with the OOD1 accuracy when predicting the *factors that are not included in the ground-truth state*, i.e., those that necessarily have to be inferred from the learned representation (see Fig. 6.4), while the correlation with the prediction performance of other factors is significantly milder. A similar result holds for OOD2 scenarios.

[4] For simplicity we could refer to the factors by some arbitrary indices such that $\mathcal{G} \subset \mathbb{N}$.

[5] Denoting by $\mathbf{y}_p$ and $\mathbf{y}_t$ the prediction targets of the proxy and target tasks, respectively, we could attempt to formalize this notion using the mutual information $I(\mathbf{y}_p; \mathbf{y}_t)$. In reinforcement learning, for example, the prediction target might be the reward (or expected discounted return) for a state–action pair or the action probabilities of an optimal stochastic policy.

CHAPTER 4

Study on object-centric representations

In this chapter, we motivate and outline the design of the study in Paper III, and discuss its main results. Further details can be found in the paper (Chapter 7) and in the corresponding supplementary material (Appendix C). The most relevant background for this chapter is in Sections 2.1 and 2.4.

4.1 Introduction

Compositional generalization is widely recognized as a fundamental issue in deep learning (Greff, Steenkiste, and Schmidhuber, 2020; Lake et al., 2017). The object-centric learning literature is mainly concerned with proposing new methods, while *assuming* that learning about objects may (at least partially) solve the issue of generalization. In Paper III (Dittadi et al., 2022a), we investigate this claim via a systematic empirical study on unsupervised object-centric learning, with a focus on image data.

4.1.1 Overview of the study design

We start by formulating three hypotheses about the unsupervised learning of object-centric representations, roughly following assumptions made in previous work but largely left implicit. In summary, an encoder that separates objects should:

1. Separately represent each object's properties in a complete and accurate manner, and therefore be useful for arbitrary downstream tasks.

2. Be robust to distribution shifts affecting a single object per image, in the sense that the representations of all other objects should still be reliable, even if the affected object has out-of-distribution properties.

3. Be robust to distribution shifts globally affecting the input, such as introducing occlusions, cropping the input image, or increasing the number of objects beyond the maximum number observed in the training distribution.

To investigate these hypotheses, we design a systematic experimental study in which we train and evaluate object-centric learning models on multi-object datasets that are annotated with segmentation masks and object properties.

Models. We focus on slot-based models, a popular approach for unsupervised object-centric learning (see Section 2.4.3), but we include models covering different approaches to object separation: instance slots (Slot Attention), sequential slots (MONet and GENESIS), and spatial slots (SPACE). Moreover, we train standard variational autoencoders (Section 2.2) in an attempt to gauge the downstream usefulness of non-object-centric representations (see Section 4.1.2).

Datasets. We train and evaluate the models on five multi-object datasets that have been used in the literature as benchmarks for object-centric models. All of them have segmentation masks; four of them have annotations for all object properties. In Fig. 4.1 (top row), we show examples from the datasets (see Fig. C.1 in Appendix C.2 for more details).

Evaluation. We use segmentation masks to directly assess how a trained model separates objects. The object property annotations allow us to evaluate representations via a downstream task that consists in predicting the ground-truth factors of variation of all objects. This task is similar to downstream prediction tasks that are common in the disentanglement literature, including Papers I and II. The general idea is that, if this information can be retrieved efficiently from the representation, it stands to reason that any arbitrary downstream task that depends on these underlying factors should also be solvable.

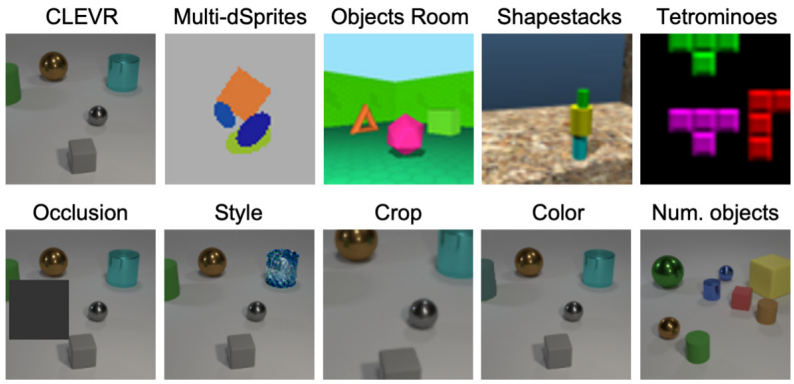


Figure 4.1. Top: examples from the five datasets considered in the study. Bottom: Example of distribution shifts applied to CLEVR. Figure from Paper III (Dittadi et al., 2022a).

Probing generalization. Finally, to assess how object separation and usefulness of the representations are affected by distribution shifts (Hypotheses 2 and 3), we evaluate segmentation accuracy and downstream task performance after introducing these shifts. In Fig. 4.1 (bottom row), we show examples of distribution shifts considered in our study (see Fig. C.2 in Appendix C.3.3 for a complete overview).

4.1.2 On the comparison with non-object-centric models

In Chapter 3 (see Papers I and II in Chapters 5 and 6) we have discussed that disentanglement may not necessarily always be useful for downstream tasks. Similarly, in Paper III, we investigate the extent to which object properties can be accurately predicted from the "flat" distributed representations of VAEs, as opposed to object-centric representations. To this end, we train VAEs with a latent space size that is comparable with the global representation size of slot-based models, and attempt a direct comparison on downstream task performance.

While it is not straightforward to gauge the fairness of this comparison, there are a few crucial points that should be discussed. The first point concerns the size of the models. We performed a light hyperparameter search (mostly on model depth) to obtain models that would at least reconstruct the input relatively well. Since some of these datasets are relatively complex—although this may not be clear from their visual appearance—the final VAEs employed in our study are relatively large.

(Interestingly, Multi-dSprites turned out to be the most challenging dataset in this hyperparameter search, possibly due to the visual complexity and heavy occlusions.) This may raise questions regarding the fairness of the comparison. On the other hand, we should note that: (i) the decoder now faces the challenge of generating the entire image at once, while typically in slot-based models a simpler decoder is applied independently to each slot to generate a single object in the scene; (ii) the object-centric models considered in this study are already rather diverse in terms of size and the largest ones are even comparable to our VAEs in terms of parameter count.

A second observation is that the downstream task typically considered in object-centric learning is in some sense meant for object-centric models. In fact, in a multi-object setting, predicting the properties of all objects is effectively a *set prediction* task. In a slot-based object-centric representation where all slots have a common format, one can apply a simple downstream model to each slot and then match the predictions to the ground-truth labels to minimize the total loss. In other words, we can directly exploit the set-like structure of the representations. Conversely, in the VAE setting, a single vector represents the entire scene, so the same setup is not directly applicable.

A possible approach to overcome this issue, as suggested by Greff et al. (2019), is to use the whole-scene representation to predict the properties of all objects at once, such that the objects are sorted lexicographically according to their properties. Potential issues with this approach are:

1. The model's input and output size are substantially different and there is no weight sharing.

2. The model must learn to sort objects, which could be relatively challenging.

3. On the other hand, since the target is sorted according to object properties, there is a bias that can be exploited by the model to predict object properties better than by random chance.

4. When testing such a model with more objects than seen during training, we are in effect probing the generalization capabilities (extrapolation, in some sense) of the model. Conversely, in the slot-based case, matching more slots to more objects has nothing to do with the downstream model, as it is simply performed by a combinatorial optimization algorithm (e.g., the Hungarian algorithm).

Another possible solution is to have the model output a flat vector containing predicted properties of all objects in no particular order, and then match the predictions to ground-truth object properties using the loss function, as we do in the slot-based case. This has similar issues and biases as the method discussed above.

In summary, when using set prediction as a downstream task, comparing object-centric and distributed representations is a challenging problem. Although we attempt to adapt this task to distributed representations in a relatively simple manner in order to minimize potential confounding effects, the comparison is arguably still not entirely fair.

4.1.3 Library

A side product of this study is a PyTorch-based (Paszke et al., 2019) Python library for training and evaluating object-centric learning methods. Since the five chosen datasets are not necessarily straightforward to use in general (e.g., the ones from the DeepMind dataset suite (Kabra et al., 2019) require TensorFlow (Abadi et al., 2016)), we repackaged all datasets into a common, general-purpose format. We adapted the official PyTorch implementations of GENESIS and SPACE to our framework, and re-implemented MONet and the Slot Attention autoencoder. Note that, since none of the models was originally tested on all the datasets considered here, some hyperparameter sweeps were necessary for some model–dataset combinations. After training a model, the library allows for automatic evaluation of segmentation metrics and reconstruction error, as well as downstream property prediction with various matching strategies and downstream models. All these evaluations can be performed out of the box on new models or datasets, as long as they implement the standard interface defined in our library.

4.2 Main results and discussion

In this section, we summarize and discuss the key findings of our study.

We find strong correlations between the ARI and the MSE across all five datasets: given a set of models trained on the same dataset, those that reconstruct the input more accurately also tend to separate objects better according to the ARI. Therefore,

the MSE can be a useful proxy metric to select high-ARI models, when ground-truth validation masks are not available. Other segmentation metrics (SC and mSC) agree with the ARI to a varying extent depending on the dataset, and their correlation with the MSE is milder and inconsistent across datasets (see Fig. 7.3). However, as discussed below, we found that the ARI appears to be a generally more useful segmentation metric than the others considered in our study.

Segmentation may not be the ultimate goal, and we might be interested in learning object-centric representations for downstream tasks (see Section 2.1.3). This is related to our first hypothesis above: object representations should contain useful information for downstream tasks, and information about different objects should be stored separately. Indeed, in our experiments, we observe that good performance on object property prediction can be achieved by downstream models that receive learned object-centric representations as input. Interestingly, we find the ARI to be the only segmentation metric that consistently has a strong correlation with downstream performance across all datasets, object properties, and downstream models. This points to the ARI as a valuable metric for model selection when ground-truth segmentation masks are available for validation. The MSE is also significantly correlated with downstream performance—which is unsurprising, considering its correlation with the ARI—but to a lesser extent and less consistently than the ARI. Although less effective than the ARI, the reconstruction error can therefore be useful for model selection when masks are not available.

Note, however, that in more realistic scenarios there may be irrelevant details in the image (possibly on the objects themselves) that are in effect nuisances with respect to the downstream tasks of interest (see the discussion on nuisance factors in Sections 2.1.1 and 2.1.4.1). In such cases, accurate reconstruction might be practically unrelated to correct object separation and downstream usefulness of the representations. In Papa, Winther, and Dittadi (2022), we confirm this hypothesis. We test different variations of MONet and Slot Attention on datasets with complex textures added to the objects via neural style transfer, and study the relationship between reconstruction quality, segmentation accuracy, and downstream performance on the same tasks considered in Paper III. We find that the ARI is still consistently correlated with downstream performance, while the MSE is not. It follows that, unsurprisingly, reconstructing detailed textures is largely irrelevant for downstream tasks that focus on higher-level object properties, while segmentation quality remains crucial. In conclusion, the ARI may be a valuable and relatively robust proxy metric for

the downstream usefulness of object-centric representations, even when objects have complex textures. Conversely, the reconstruction MSE is not necessarily informative when objects have a more complex appearance such as in the presence of diverse textures (Papa, Winther, and Dittadi, 2022).

Next, we are interested in comparing the downstream performance attainable from object-centric representations and from "flat", distributed ones. As discussed in Section 4.1.2, making a fair comparison is challenging, since the set prediction task considered here is particularly well-suited for set-like representations in the first place. In order to make a sensible assessment of distributed representations, we set up simple experiments where the downstream predictions are initialized to a random vector of object properties for all objects in the scene, and are optimized with the standard training procedure for downstream tasks for distributed representations. We find that the baseline performance for distributed representations is often remarkably higher than the standard "random guess" baseline we would use for object-centric representations (although in some cases it is approximately equal). Despite this advantage, downstream predictors from object-centric representations tend to outperform those from distributed representations. Although the task may arguably be more difficult for a single downstream MLP that has to predict all objects at once (see Section 4.1.2), we also observe that the performance typically does not improve significantly when increasing model size up to three hidden layers.

Our second hypothesis is that, when one object is out of distribution, the representations of other objects should be robust, i.e., they should still faithfully represent those objects' properties. First, we observe that the models tend to be able to segment the scene correctly in this single-OOD-object scenario. Then, we focus on the representations of the objects in the scene, and use the downstream prediction performance to assess how they are affected by distribution shifts. Our results, presented in Section 7.4.3, indicate that:

- Property prediction for out-of-distribution objects deteriorates. Moreover, the representations of the out-of-distribution objects are not reliable, since even retraining the downstream model after the distribution shift has occurred does not significantly improve performance.

- The downstream model can correctly predict the properties of the in-distribution objects, as if the distribution shift did not occur at all—this suggests that the representations of the in-distribution objects are largely unaffected by one ob-

ject being out of distribution.

Our third hypothesis is that object-centric models are robust even when a distribution shifts affects global properties of the scene. First, we look for a potential degradation of segmentation performance at test time when the data undergoes a distribution shift of this type. Unlike for single-object shifts, for global shifts we observe varying results depending on the specific shift. For example, cropping and enlarging the scene often harms segmentation quality, while occlusions appear to have a minor effect. Interestingly, when the number of objects increases at test time (in CLEVR), we see a relatively small drop in the ARI for object-centric models, and the reconstruction error increases less significantly than for the vanilla VAE models. This is presumably due to the strong inductive bias of object-centric models towards treating objects separately. However, what may seem surprising is that VAEs trained with up to 6 objects can generate rather coherent samples that contain more. This suggests that they might be capable of representing more objects than observable in the training distribution. In fact, given an image with more objects they seem to be able to reconstruct the correct number of objects, as shown in Fig. 4.2. On the other hand, some of these objects have incorrect properties, suggesting that this extrapolation behaviour—where object-centric models have a clear advantage by design—is limited. We present a quantitative analysis in Paper III (Chapter 7), and Appendix C.4 includes additional qualitative and quantitative results.

Regarding the informativeness and usefulness of the object representations under global distribution shifts, our findings are mostly negative: in all datasets and for all models, the prediction performance deteriorates significantly for most object properties (although to varying degrees), and cannot be recovered even by adjusting the downstream model post hoc on the OOD data.

A point that merits discussion is that, while the definition of single-object shifts is clear—one object is OOD an the others are ID—it is not as straightforward to characterize *global* distribution shifts. *Cropping* is essentially equivalent to an object-wise distribution shift that enlarges all objects. On the other hand, if we accept this interpretation, we also have to view this distribution shift as affecting the number of objects in the scene, their relative spatial arrangement (their centers are farther apart), and the fact that, on average, more objects than usual may be partially out of the field of view. Finally, note that cropping may be more simply seen as a shift in a global property of the scene such as camera view—from this point of view, it is more easily interpreted as global shift. *Occlusion*, another global distribution shift in

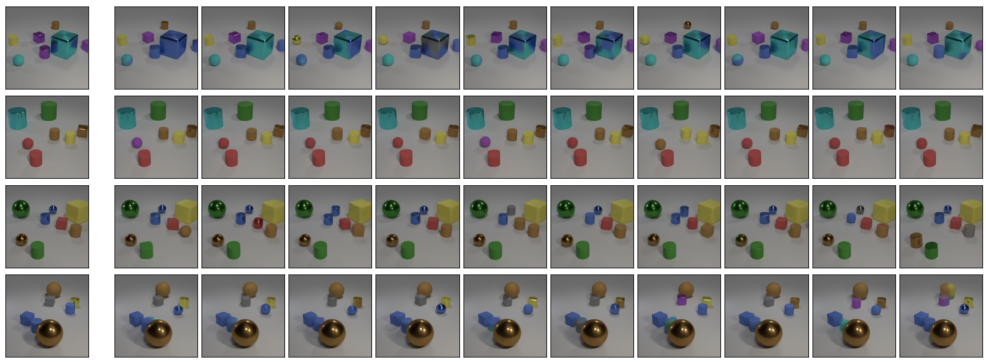


Figure 4.2. Input and reconstructions of 4 randomly selected CLEVR images with more than 6 objects by VAEs trained with up to 6 objects. Leftmost column: input image. Other columns: reconstructions by the 10 VAEs in our study that were trained on CLEVR6. Note that some of the variability comes from sampling $\mathbf{z}$ rather than using the posterior mean (which is typically used as representation in VAEs), but this does not explain that a few object properties are sometimes inferred incorrectly. On the other hand, it is worth noting that all VAEs seem to be able to produce coherent samples with more objects than observed during training.

our categorization, could also be viewed as adding a new object that is very different from any previously seen object, while leaving other objects untouched. From this perspective, it is not particularly surprising that object-centric models seem to exhibit at least partial robustness to this shift. Finally, the *number of objects* can easily be interpreted as a global property of the scene. However, considering one object at a time, there is in fact no distribution shift at all. This partly explains why some object-centric models tend to be relatively robust to this distribution shift. On the other hand, most of these models still have to compute visual features and, most importantly, some form of attention mask, from the entire image: this, following the argument on OOD2 generalization from Chapter 3 and Papers I and II, may explain the minor drop in performance.

In fact, *all* distribution shifts considered in this study bring the encoders out of distribution, and therefore correspond to the OOD2 setting in Papers I and II. However, the fact that both segmentation and representation quality are affected to a widely varying degree suggests that there might be inductive biases in the models that make representation functions more robust to some shift types than to others. In particular, object-centric models do not appear to be significantly affected when only one object is out of distribution. In future work, it would be relevant to investigate this further

by comparing the generalization of object-centric models and other models in a fairer way, and by testing a broader range of distribution shifts.

On a related note, it would be interesting to consider the OOD1 scenario from Papers I and II: given a class of representation functions trained on a large enough data distribution (such that they are never out of distribution at test time), do downstream models generalize better if the representations are object-centric? For example, we could learn representation functions on the full CLEVR dataset, and then train downstream models on a subset of CLEVR that contains few objects, or on another subset that does not contain red objects. We would then be able to investigate which type of representation shows more potential for the OOD generalization of downstream models. Arguably, this is in fact the flavor of generalization that should be enabled by structured, or even causal, representations of the data. What good is a representation function that perfectly inverts the data generating process if we use it out of distribution, where it no longer works reliably?

Another type of distribution shift we are not considering in this study is a shift in correlations between factors of variation. One example is a correlation between factors within each object, as we have explored in Träuble et al. (2021) for disentanglement learning. In the multi-object setting, a similar correlation has been investigated by Greff et al. (2019), who test IODINE on held-out images from CLEVR containing green spheres. It would be interesting to test the robustness of object-centric models to this shift, and study whether per-object disentanglement plays a role (see the generalization results in Träuble et al. (2021)). Another example that specifically applies to the multi-object setting is object co-occurrence. E.g., if all images either have zero or two red spheres, will an object-centric model represent two red spheres in the same slot? Should it not? Will it be able to generalize to images that only show one red sphere? This scenario, which includes the special case of object–part hierarchies, also brings up the question of how the notion of object should be defined (Greff, Steenkiste, and Schmidhuber, 2020).

Beside distribution shifts, further considerations should be made on the different ways models and representations could be evaluated. While a clear advantage of object-centric representations is the possibility of performing separate interventions on single objects,[1] this is in fact not feasible in a sensible, coherent manner in many existing object-centric models (e.g., MONet blends generated objects into a single

[1] This includes, for example, performing latent traversals on one slot at a time, or replacing two slots with a convex combination of the two to smoothly interpolate between them.

image using alpha masks that are inferred directly from the input). In addition, as discussed in Section 4.1.2, although factor prediction is a reasonable proxy task, testing learned representations on more complex and practically relevant downstream tasks (e.g., relational question answering or reinforcement learning) would improve our understanding of models trained on multi-object data.

We have also observed that distributed representations (in our case, obtained by training standard VAEs) might in fact be more useful than expected, but we could not draw definitive conclusions because of the fundamental incompatibility of these representations with the set prediction task in our study. Moreover, we found object-centric models to be rather sensitive to hyperparameters and random seeds, probably because they are structured models with discrete components. When the optimization goes wrong, and a model does not segment the scene in a meaningful way, the suitability for downstream tasks may decrease drastically. By contrast, although simpler models like VAEs lack some of the beneficial inductive biases of object-centric models, they are generally easier to optimize and require less hyperparameter tuning. Especially in light of the recent progress made by focusing on scaling and engineering rather than methodology, an open question is whether these simpler models may indeed be sufficient, given enough data and large enough models. For these reasons, a more thorough comparison of object-centric and distributed representations is needed. New downstream tasks should ideally make this comparison fairer and sounder, and further inform future research efforts.

CHAPTER 5

Paper I: On the Transfer of Disentangled Representations in Realistic Settings

Authors: Andrea Dittadi,* Frederik Träuble,* Francesco Locatello, Manuel Wüthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, Bernhard Schölkopf.
*Equal contribution.

Status: Published in *International Conference on Learning Representations*, 2021.

Abstract. Learning meaningful representations that disentangle the underlying structure of the data generating process is considered to be of key importance in machine learning. While disentangled representations were found to be useful for diverse tasks such as abstract reasoning and fair classification, their scalability and real-world impact remain questionable. We introduce a new high-resolution dataset with 1M simulated images and over 1,800 annotated real-world images of the same setup. In contrast to previous work, this new dataset exhibits correlations, a complex underlying structure, and allows to evaluate transfer to unseen simulated and real-world settings where the encoder i) remains in distribution or ii) is out of distribution. We propose new architectures in order to scale disentangled representation learning to realistic high-resolution settings and conduct a large-scale empirical study of disentangled representations on this dataset. We observe that disentanglement is a good predictor for out-of-distribution (OOD) task performance.

5.1 Introduction

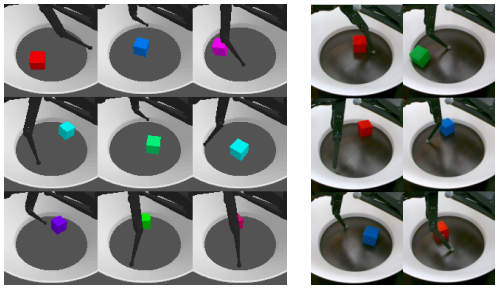


Figure 5.1. Images from the simulated dataset (left) and the real-world setup (right).

Disentangled representations hold the promise of generalization to unseen scenarios (Higgins et al., 2017b), increased interpretability (Adel, Ghahramani, and Weller, 2018; Higgins et al., 2018) and faster learning on downstream tasks (Locatello et al., 2019a; Steenkiste et al., 2019). However, most of the focus in learning disentangled representations has been on small synthetic datasets whose ground truth factors exhibit perfect independence by design. More realistic settings remain largely unexplored. We hypothesize that this is because real-world scenarios present several challenges that have not been extensively studied to date. Important challenges are scaling (much higher resolution in observations and factors), occlusions, and correlation between factors. Consider, for instance, a robotic arm moving a cube: Here, the robot arm can occlude parts of the cube, and its end-effector position exhibits correlations with the cube's position and orientation, which might be problematic for common disentanglement learners (Träuble et al., 2021). Another difficulty is that we typically have only limited access to ground truth labels in the real world, which requires robust frameworks for model selection when no or only weak labels are available.

The goal of this work is to provide a path towards disentangled representation learning in realistic settings. First, we argue that this requires a new dataset that captures the challenges mentioned above. We propose a dataset consisting of simulated observations from a scene where a robotic arm interacts with a cube in a stage (see Fig. 5.1). This setting exhibits correlations and occlusions that are typical in real-world robotics. Second, we show how to scale the architecture of disentanglement methods to perform well on this dataset. Third, we extensively analyze the usefulness of disentangled representations in terms of out-of-distribution downstream generalization, both in terms of held-out factors of variation and sim-to-real transfer. In fact, our dataset is based on the TriFinger robot from Wüthrich et al. (2020), which can be built to test the deployment of models in the real world. While the analysis in this paper focuses on the transfer and generalization of predictive models, we hope that our dataset may

serve as a benchmark to explore the usefulness of disentangled representations in real-world control tasks.

The contributions of this paper can be summarized as follows:

- We propose a new dataset for disentangled representation learning, containing 1M simulated high-resolution images from a robotic setup, with seven partly correlated factors of variation. Additionally, we provide a dataset of over 1,800 annotated images from the corresponding real-world setup that can be used for challenging sim-to-real transfer tasks. These datasets are made publicly available.[1]

- We propose a new neural architecture to successfully scale VAE-based disentanglement learning approaches to complex datasets.

- We conduct a large-scale empirical study on generalization to various transfer scenarios on this challenging dataset. We train 1,080 models using state-of-the-art disentanglement methods and discover that disentanglement is a good predictor for out-of-distribution (OOD) performance of downstream tasks.

5.2 Related Work

Disentanglement methods. Most state-of-the-art methods for disentangled representation learning are based on the framework of variational autoencoders (VAEs) (Kingma and Welling, 2014; Rezende, Mohamed, and Wierstra, 2014). A (high-dimensional) observation $\mathbf{x}$ is assumed to be generated according to the latent variable model $p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$ where the latent variables $\mathbf{z}$ have a fixed prior $p(\mathbf{z})$. The generative model $p_\theta(\mathbf{x}|\mathbf{z})$ and the approximate posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$ are typically parameterized by neural networks, which are optimized by maximizing the evidence lower bound (ELBO):

$$\mathcal{L}_{VAE} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) \leq \log p(\mathbf{x}) \tag{5.1}$$

As the above objective does not enforce any structure on the latent space except for some similarity to $p(\mathbf{z})$, different regularization strategies have been proposed, along

[1] http://people.tuebingen.mpg.de/ei-datasets/iclr_transfer_paper/robot_finger_datasets.tar (6.18 GB)

with evaluation metrics to gauge the disentanglement of the learned representations (Burgess et al., 2018; Chen et al., 2018; Eastwood and Williams, 2018; Higgins et al., 2017a; Kim and Mnih, 2018; Kumar, Sattigeri, and Balakrishnan, 2018). Recently, Locatello et al. (2019b, Theorem 1) showed that the purely unsupervised learning of disentangled representations is impossible. This limitation can be overcome without the need for explicitly labeled data by introducing weak labels (Locatello et al., 2020b; Shu et al., 2020). Ideas related to disentangling the factors of variation date back to the non-linear ICA literature (Bach and Jordan, 2002; Comon, 1994; Gresele et al., 2019; Hyvarinen and Morioka, 2016; Hyvarinen, Sasaki, and Turner, 2019; Hyvärinen and Pajunen, 1999; Jutten and Karhunen, 2003). Recent work combines non-linear ICA with disentanglement (Khemakhem et al., 2020; Klindt et al., 2020; Sorrenson, Rother, and Köthe, 2020).

Evaluating disentangled representations. The *BetaVAE* (Higgins et al., 2017a) and *FactorVAE* (Kim and Mnih, 2018) scores measure disentanglement by performing an intervention on the factors of variation and predicting which factor was intervened on. The *Mutual Information Gap (MIG)* (Chen et al., 2018), *Modularity* (Ridgeway and Mozer, 2018), *DCI Disentanglement* (Eastwood and Williams, 2018) and *SAP* scores (Kumar, Sattigeri, and Balakrishnan, 2018) are based on matrices relating factors of variation and codes (e.g. pairwise mutual information, feature importance and predictability).

Datasets for disentanglement learning. *dSprites* (Higgins et al., 2017a), which consists of binary low-resolution 2D images of basic shapes, is one of the most commonly used synthetic datasets for disentanglement learning. *Color-dSprites*, *Noisy-dSprites*, and *Scream-dSprites* are slightly more challenging variants of dSprites. The *SmallNORB* dataset contains toy images rendered under different lighting conditions, elevations and azimuths (LeCun, Huang, and Bottou, 2004). *Cars3D* (Reed et al., 2015) exhibits different car models from Fidler, Dickinson, and Urtasun (2012) under different camera viewpoints. *3dshapes* is a popular dataset of simple shapes in a 3D scene (Kim and Mnih, 2018). Finally, Gondal et al. (2019) proposed *MPI3D*, containing images of physical 3D objects with seven factors of variation, such as object color, shape, size and position available in a simulated, simulated and highly realistic rendered simulated variant. Except MPI3D which has over 1M images, the size of the other datasets is limited with only $17,568$ to $737,280$ images. All of the above datasets exhibit perfect independence of all factors, the number of possible states is on the order of 1M or less, and due to their static setting they do not allow for

dynamic downstream tasks such as reinforcement learning. In addition, except for SmallNORB, the image resolution is limited to 64x64 and there are no occlusions.

Other related work. Locatello et al. (2020b) probed the out-of-distribution generalization of downstream tasks trained on disentangled representations. However, these representations are trained on the entire dataset. Generalization and transfer performance especially for representation learning has likewise been studied in Arjovsky et al. (2019), Dayan (1993), Gowal et al. (2020), Heinze-Deml and Meinshausen (2017), Krueger et al. (2020), Li et al. (2018), Muandet, Balduzzi, and Schölkopf (2013), Rojas-Carulla et al. (2018), and Suter et al. (2019). For the role of disentanglement in causal representation learning we refer to the recent overview by Schölkopf et al. (2021). Träuble et al. (2021) systematically investigated the effects of correlations between factors of variation on disentangled representation learners. Transfer of learned disentangled representations from simulation to the real world has been recently investigated by Gondal et al. (2019) on the MPI3D dataset, and previously by Higgins et al. (2017b) in the context of reinforcement learning. Sim-to-real transfer is of major interest in the robotic learning community, because of limited data and supervision in the real world (Andrychowicz et al., 2020; James et al., 2019; Peng et al., 2018; Rusu et al., 2017; Tobin et al., 2017; Yan et al., 2020).

Table 5.1. Factors of variation in the proposed dataset. Values are linearly spaced in the specified intervals. Joint angles are in radians, cube positions in meters.

FoV	**Values**
Upper joint	30 values in $[-0.65, +0.65]$
Middle joint	30 values in $[-0.5, +0.5]$
Lower joint	30 values in $[-0.8, +0.8]$
Cube position x	30 values in $[-0.11, +0.11]$
Cube position y	30 values in $[-0.11, +0.11]$
Cube rotation	10 values in $[0°, 81°]$
Cube color hue	12 values in $[0°, 330°]$

5.3 Scaling Disentangled Representations to Complex Scenarios

A new challenging dataset. Simulated images in our dataset are derived from the trifinger robot platform introduced by Wüthrich et al. (2020). The motivation for choosing this setting is that (1) it is challenging due to occlusions, correlations, and other difficulties encountered in robotic settings, (2) it requires modeling of fine details such as tip links at high resolutions, and (3) it corresponds to a robotic setup, so that learned representations can be used for control and reinforcement learning in simulation and in the real world. The scene comprises a robot finger with three joints that can be controlled to manipulate a cube in a bowl-shaped stage. Fig. 5.1 shows examples of scenes from our dataset. The data is generated from 7 different factors of variation (FoV) listed in Table 5.1. Unlike in previous datasets, not all FoVs are independent: The end-effector (the tip of the finger) can collide with the floor or the cube, resulting in infeasible combinations of the factors (see Appendix A.2.1). We argue that such correlations are a key feature in real-world data that is not present in existing datasets. The high FoV resolution results in approximately 1.52 billion feasible states, but the dataset itself only contains one million of them (approximately 0.065% of all possible FoV combinations), realistically rendered into 128×128 images. Additionally, we recorded an annotated dataset under the same conditions in the real-world setup: we acquired 1,809 camera images from the same viewpoint and recorded the labels of the 7 underlying factors of variation. This dataset can be used for out-of-distribution evaluations, few-shot learning, and testing other sim-to-real aspects.

Model architecture. When scaling disentangled representation learning to more complex datasets, such as the one proposed here, one of the main bottlenecks in current VAE-based approaches is the flexibility of the encoder and decoder networks. In particular, using the architecture from Locatello et al. (2019b), none of the models we trained correctly captured all factors of variation or yielded high-quality reconstructions. While the increased image resolution already presents a challenge, the main practical issue in our new dataset is the level of detail that needs to be modeled. In particular, we identified the cube rotation and the lower joint position to be the factors of variation that were the hardest to capture. This is likely because these factors only produce relatively small changes in the image and hence the reconstruction error.

To overcome these issues, we propose a deeper and wider neural architecture than those commonly used in the disentangled representation learning literature, where the

encoder and decoder typically have 4 convolutional and 2 fully-connected layers. Our encoder consists of a convolutional layer, 10 residual blocks, and 2 fully-connected layers. Some residual blocks are followed by 1x1 convolutions that change the number of channels, or by average pooling that downsamples the tensors by a factor of 2 along the spatial dimensions. Each residual block consists of two 3x3 convolutions with a leaky ReLU nonlinearity, and a learnable scalar gating mechanism (Bachlechner et al., 2020). Overall, the encoder has 23 convolutional layers and 2 fully connected layers. The decoder mirrors this architecture, with average pooling replaced by bilinear interpolation for upsampling. The total number of parameters is approximately 16.3M. See Appendix A.1 for further implementation details.

Experimental setup. We perform a large-scale empirical study on the simulated dataset introduced above by training 1,080 β-VAE models.[2] For further experimental details we refer the reader to Appendix A.1. The hyperparameter sweep is defined as follows:

- We train the models using either unsupervised learning or weakly supervised learning (Locatello et al., 2020b). In the weakly supervised case, a model is trained with pairs of images that differ in k factors of variation. Here we fix $k = 1$ as it was shown to lead to higher disentanglement by Locatello et al. (2020b). The dataset therefore consists of 500k pairs of images that differ in only one FoV.

- We vary the parameter β in $\{1, 2, 4\}$, and use linear deterministic warm-up (Bowman et al., 2015; Sønderby et al., 2016) over the first $\{0, 10000, 50000\}$ training steps.

- The latent space dimensionality is in $\{10, 25, 50\}$.

- Half of the models are trained with additive noise in the input image. This choice is motivated by the fact that adding noise to the input of neural networks has been shown to be beneficial for out-of-distribution generalization (Bishop, 1995; Sietsma and Dow, 1991).

- Each of the 108 resulting configurations is trained with 10 random seeds.

[2]Training these models requires approximately 2.8 GPU years on NVIDIA Tesla V100 PCIe.

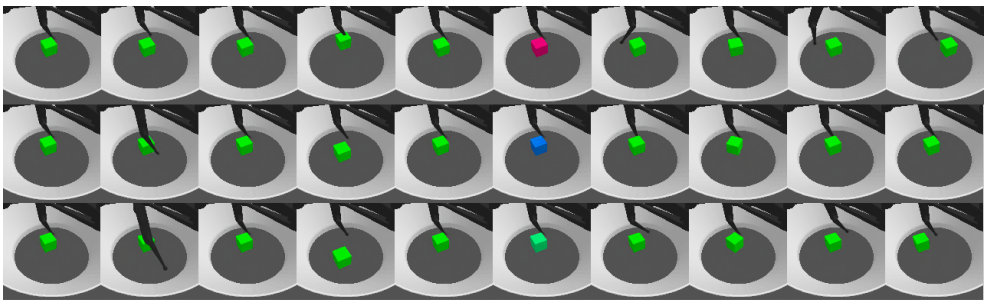


Figure 5.2. Latent traversals of a trained model that perfectly disentangles the dataset's FoVs. In each column, all latent variables but one are fixed.

Can we scale up disentanglement learning? Most of the trained VAEs in our empirical study fully capture all the elements of a scene, correctly model heavy occlusions, and generate detailed, high-quality samples and reconstructions (see Appendix A.2.2). From visual inspections such as the latent traversals in Fig. 5.2, we observe that many trained models fully disentangle the ground-truth factors of variation. This, however, appears to only be possible in the weakly supervised scenario. The fact that models trained without supervision learn entangled representations is in line with the impossibility result for the unsupervised learning of disentangled representations from Locatello et al. (2019b). Latent traversals from a selection of models with different degrees of disentanglement are presented in Appendix A.2.3. Interestingly, the high-disentanglement models seem to correct for correlations and interpolate infeasible states, i.e. the fingertip traverses through the cube or the floor.

Summary: The proposed architecture can scale disentanglement learning to more realistic settings, but a form of weak supervision is necessary to achieve high disentanglement.

How useful are common disentanglement metrics in realistic scenarios? The violin plot in Fig. 5.3 (left) shows that DCI and MIG measure high disentanglement under weak supervision and lower disentanglement in the unsupervised setting. This is consistent with our qualitative conclusion from visual inspection of the models (Appendix A.2.3) and with the aforementioned impossibility result. Many of the models trained with weak supervision exhibit a very high DCI score (29% of them have >99% DCI, some of them up to 99.89%). SAP and Modularity appear to be ineffective at capturing disentanglement in this setting, as also observed by Locatello et al.

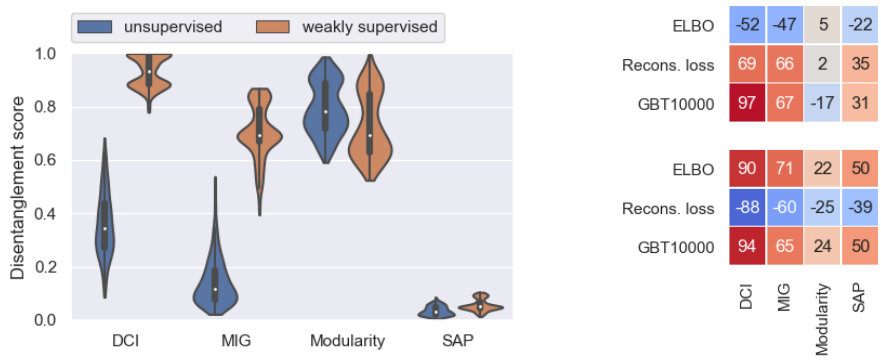


Figure 5.3. **Left**: Disentanglement metrics aggregating all hyperparameters except for supervision type. **Right**: Spearman Rank correlations of the disentanglement metrics with the ELBO, the reconstruction loss, and the test error of a GBT classifier trained on 10,000 labelled data points. The upper rank correlations correspond to the unsupervised models and the lower ones to the weakly supervised models.

(2019b). Finally, note that the BetaVAE and FactorVAE metrics are not straightforward to be evaluated on datasets that do not contain all possible combinations of factor values. According to Fig. 5.3 (right), DCI and MIG strongly correlate with test accuracy of GBT classifiers predicting the FoVs. In the weakly supervised setting, these metrics are strongly correlated with the ELBO (positively) and with the reconstruction loss (negatively). We illustrate these relationships in more detail in Appendix A.2.4. Such correlations were also observed by Locatello et al. (2020b) on significantly less complex datasets, and can be exploited for unsupervised model selection: these unsupervised metrics can be used as proxies for disentanglement metrics, which would require fully labeled data.

Summary: DCI and MIG appear to be useful disentanglement metrics in realistic scenarios, whereas other metrics seem to fall short of capturing disentanglement or can be difficult to compute. When using weak supervision, we can select disentangled models with unsupervised metrics.

5.4 Framework for the Evaluation of OOD Generalization

Previous work has focused on evaluating the usefulness of disentangled representations for various downstream tasks, such as predicting ground truth factors of variation, fair classification, and abstract reasoning. Here we propose a new framework for evaluating the out-of-distribution (OOD) generalization properties of representations. More specifically, we consider a downstream task—in our case, regression of ground truth factors—trained on a learned representation of the data, and evaluate the performance on a held-out test set. While the test set typically follows the same distribution as the training set (in-distribution generalization), we also consider test sets that follow a different distribution (out-of-distribution generalization). Our goal is to investigate to what extent, if at all, downstream tasks trained on disentangled representations exhibit a higher degree of OOD generalization than those trained on entangled representations.

Let $\mathcal{D}$ denote the training set for disentangled representation learning. To investigate OOD generalization, we train downstream regression models on a subset $\mathcal{D}_1 \subset \mathcal{D}$ to predict ground truth factor values from the learned representation computed by the encoder. We independently train one predictor per factor. We then test the regression models on a set $\mathcal{D}_2$ that differs distributionally from the training set $\mathcal{D}_1$, as it either contains images corresponding to held-out values of a chosen FoV (e.g. unseen object colors), or it consists of real-world images. We now differentiate between two scenarios: (1) $\mathcal{D}_2 \subset \mathcal{D}$, i.e. the OOD test set is a subset of the dataset for representation learning; (2) $\mathcal{D}$ and $\mathcal{D}_2$ are disjoint and distributionally different. These two scenarios will be denoted by *OOD1* and *OOD2*, respectively. For example, consider the case in which distributional shifts are based on one FoV: the color of the object. Then, we could define these datasets such that images in $\mathcal{D}$ always contain a red or blue object, and those in $\mathcal{D}_1 \subset \mathcal{D}$ always contain a red object. In the OOD1 scenario, images in $\mathcal{D}_2$ would always contain a blue object, whereas in the OOD2 case they would always contain an object that is neither red nor blue.

The regression models considered here are Gradient Boosted Trees (GBT), random forests, and MLPs with $\{1, 2, 3\}$ hidden layers. Since random forests exhibit a similar behavior to GBTs, and all MLPs yield similar results to each other, we choose GBTs and the 2-layer MLP as representative models and only report results for those. To quantify prediction quality, we normalize the ground truth factor values to the range $[0, 1]$, and compute the mean absolute error (MAE). Since the values are normalized,

we can define our transfer metric as the average of the MAE over all factors (except for the FoV that is OOD).

5.5 Benefits and Transfer of Structured Representations

Experimental setup. We evaluate the transfer metric introduced in Section 5.4 across all 1,080 trained models. To compute this metric, we train regression models to predict the ground truth factors of variation, and test them under distributional shift. We consider distributional shifts in terms of cube color or sim-to-real, and we do not evaluate downstream prediction of cube color. We report scores for two different regression models: a Gradient Boosted Tree (GBT) and an MLP with 2 hidden layers of size 256. In Appendix A.1 we provide details on the datasets used in this section.

In the OOD1 setting, we have $\mathcal{D}_2 \subset \mathcal{D}$, hence the encoder is in-distribution: we are testing the predictor on representations of images that were in the training set of the representation learning algorithm. Therefore, we expect the representations to be meaningful. We consider three scenarios:

- OOD1-A: The regression models are trained on 1 cube color (red) and evaluated on the remaining 7 colors.
- OOD1-B: The regression models are trained on 4 cube colors with high hue in the HSV space, and evaluated on 4 cube colors with low hue (extrapolation).
- OOD1-C: The regression models are again trained and evaluated on 4 cube colors, but the training and evaluation colors are alternating along the hue dimension (interpolation).

In the more challenging setting where even the encoder is out-of-distribution (OOD2, with $\mathcal{D}_2 \cap D = \varnothing$), we train the regression models on a subset of the training set $\mathcal{D}$ that includes all 8 cube colors, and we consider the two following scenarios:

- OOD2-A: The regression models are evaluated on simulated data, on 4 cube colors that are out of the encoder's training distribution.
- OOD2-B: The regression models are evaluated on real-world images of the robotic setup, without any adaptation or fine-tuning.

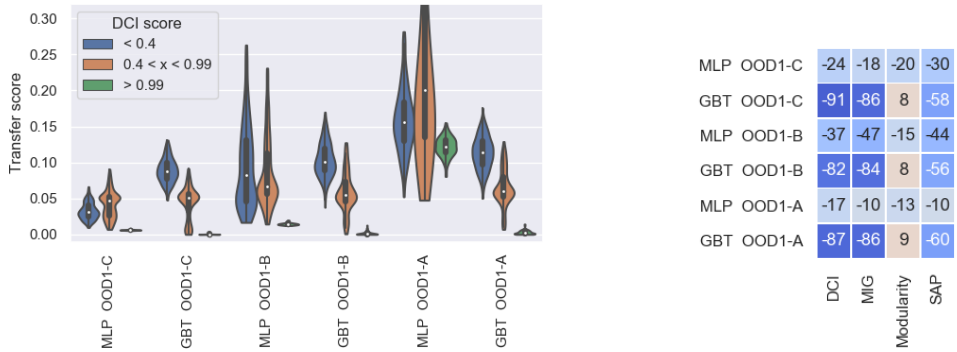


Figure 5.4. Higher disentanglement corresponds to better generalization across all OOD1 scenarios, as seen from the transfer scores (left). The transfer score is computed as the mean absolute prediction error of ground truth factor values (lower is better). This correlation is particularly evident in the GBT case, whereas MLPs appear to exhibit better OOD1 transfer with very high disentanglement only. These results are mirrored in the Spearman rank correlations between transfer scores and disentanglement metrics (right).

Is disentanglement correlated with OOD1 generalization? In Fig. 5.4 we consistently observe a negative correlation between disentanglement and transfer error across all OOD1 settings. The correlation is mild when using MLPs, strong when using GBTs. This difference is expected, as GBTs have an axis-alignment bias whereas MLPs can—given enough data and capacity—disentangle an entangled representation more easily. Our results therefore suggest that **highly disentangled representations are useful for generalizing out-of-distribution as long as the encoder remains in-distribution**. This is in line with the correlation found by Locatello et al. (2019b) between disentanglement and the GBT10000 metric. There, however, GBTs are tested on the same distribution as the training distribution, while here we test them under distributional shift. Given that the computation of disentanglement scores requires labels, this is of little benefit in the unsupervised setting. However, it can be exploited in the weakly supervised setting, where disentanglement was shown to correlate with ELBO and reconstruction loss (Section 5.3). Therefore, model selection for representations that transfer well in these scenarios is feasible based on the ELBO or reconstruction loss, when weak supervision is available. Note that, in absolute terms, the OOD generalization error with encoder in-distribution (OOD1) is very low in the high-disentanglement case (the only exception being the MLP in the OOD1-C case, with the 1-7 color split, which seems to overfit). This suggests that disentangled representations can be useful in downstream tasks even when transferring

out of the training distribution.

Summary: Disentanglement seems to be positively correlated with OOD generalization of downstream tasks, provided that the encoder remains in-distribution (OOD1). Since in the weakly supervised case disentanglement correlates with the ELBO and the reconstruction loss, model selection can be performed using these metrics as proxies for disentanglement. These metrics have the advantage that they can be computed without labels, unlike disentanglement metrics.

Is disentanglement correlated with OOD2 generalization? As we can observe in Fig. 5.5, the negative correlation between disentanglement and GBT transfer error is weaker when the encoder is out of distribution (OOD2). Nonetheless, we observe a non-negligible correlation for GBTs in the OOD2-A case, where we investigate out-of-distribution generalization along one FoV, with observations in $\mathcal{D}_2$ still generated from the same simulator. In the OOD2-B setting, where the observations are taken from cameras in the corresponding real-world setting, the correlation between disentanglement and transfer performance appears to be minor at best. This scenario can be considered a variant of zero-shot sim-to-real generalization.

Summary: Disentanglement has a minor effect on out-of-distribution generalization outside of the training distribution of the encoder (OOD2).

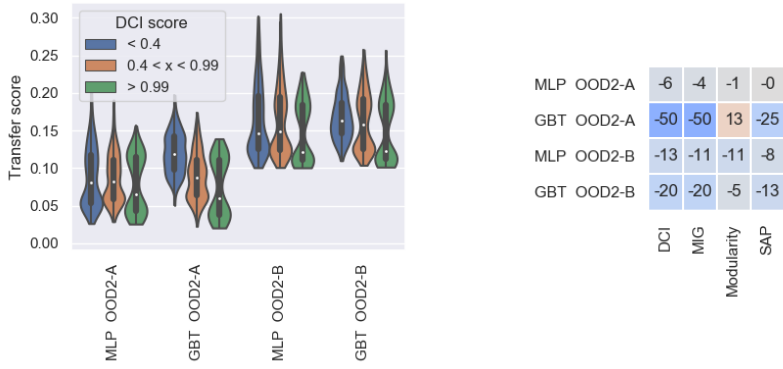


Figure 5.5. Disentanglement affects generalization across the OOD2 scenarios only minimally as seen from transfer scores (left) and corresponding rank correlations with disentanglement metrics (right).

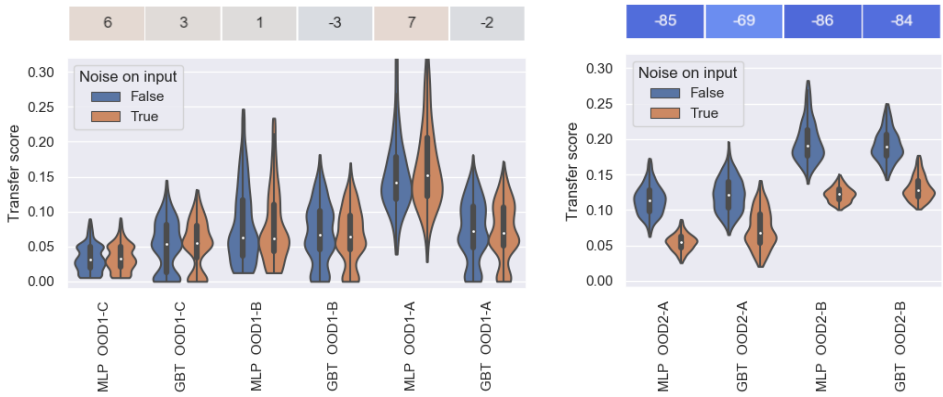


Figure 5.6. Noise improves generalization across the OOD2 scenarios and less so for the OOD1 scenarios as seen from the transfer scores. Top row: Spearman rank correlation coefficients between transfer metrics and presence of noise in the input.

What else matters for OOD2 generalization? Results in Fig. 5.6 suggest that adding Gaussian noise to the input during training as described in Section 5.3 leads to significantly better OOD2 generalization, and has no effect on OOD1 generalization. Adding noise to the input of neural networks is known to lead to better generalization (Bishop, 1995; Sietsma and Dow, 1991). This is in agreement with our results, since OOD1 generalization does not require generalization of the encoder, while OOD2 does. Interestingly, closer inspection reveals that the contribution of different factors of variation to the generalization error can vary widely. See Appendix A.2.5 for further details. In particular, with noisy input, the position of the cube is predicted accurately even in real-world images (<5% mean absolute error on each axis). This is promising for robotics applications, where the true state of the joints is observable but inference of the cube position relies on object tracking methods. Fig. 5.7 shows an example of real-world inputs and reconstructions of their simulated equivalents.

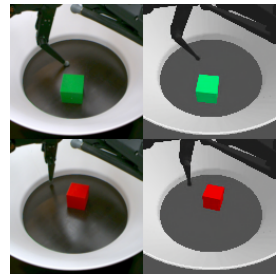


Figure 5.7. Zero-shot transfer of our models trained in simulation to real images. Left: input; right: reconstruction.

Summary: Adding input noise during training appears to be significantly beneficial for OOD2 generalization, while having no effect when the encoder is kept in its training distribution (OOD1).

5.6 Conclusion

Despite the growing importance of the field and the potential societal impact in the medical domain (Chartsias et al., 2018) and fair decision making (Locatello et al., 2019a), state-of-the-art approaches for learning disentangled representations have so far only been systematically evaluated on synthetic toy datasets. Here we introduced a new high-resolution dataset with 1M simulated images and over 1,800 annotated real-world images of the same setup. This dataset exhibits a number of challenges and features which are not present in previous datasets: it contains correlations between factors, occlusions, a complex underlying structure, and it allows for evaluation of transfer to unseen simulated and real-world settings. We proposed a new VAE architecture to scale disentangled representation learning to this realistic setting and conducted a large-scale empirical study of disentangled representations on this dataset. We discovered that disentanglement is a good predictor of OOD generalization of downstream tasks and showed that, in the context of weak supervision, model selection for good OOD performance can be based on the ELBO or the reconstruction loss, which are accessible without explicit labels. Our setting allows for studying a wide variety of interesting downstream tasks in the future, such as reinforcement learning or learning a dynamics model of the environment. Finally, we believe that in the future it will be important to take further steps in the direction of this paper by considering settings with even more complex structures and stronger correlations between factors.

Acknowledgements

The authors thank Shruti Joshi and Felix Widmaier for their useful comments on the simulated setup, Anirudh Goyal for helpful discussions and comments, and CIFAR for the support. We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Frederik Träuble.

CHAPTER 6

Paper II: The Role of Pretrained Representations for the OOD Generalization of Reinforcement Learning Agents

---

Authors: Andrea Dittadi,* Frederik Träuble,* Manuel Wüthrich, Felix Widmaier, Peter Gehler, Ole Winther, Francesco Locatello, Olivier Bachem, Bernhard Schölkopf, Stefan Bauer.

*Equal contribution (the order was chosen at random and may be changed).

Status: Published in *International Conference on Learning Representations*, 2022.

Abstract. Building sample-efficient agents that generalize out-of-distribution (OOD) in real-world settings remains a fundamental unsolved problem on the path towards achieving higher-level cognition. One particularly promising approach is to begin with low-dimensional, pretrained representations of our world, which should facilitate efficient downstream learning and generalization. By training 240 representations and over 10,000 reinforcement learning (RL) policies on a simulated robotic setup, we evaluate to what extent different properties of pretrained VAE-based representations affect the OOD generalization of downstream agents. We observe that many agents are surprisingly robust to realistic distribution shifts, including the challenging sim-to-real case. In addition, we find that the generalization performance of a simple downstream proxy task reliably predicts the generalization performance of our RL agents under a wide range of OOD settings. Such proxy tasks can thus be used to select pretrained representations that will lead to agents that generalize.

6.1 Introduction

Robust out-of-distribution (OOD) generalization is one of the key open challenges in machine learning. This is particularly relevant for the deployment of ML models to the real world, where we need systems that generalize beyond the i.i.d. (independent and identically distributed) data setting (Azulay and Weiss, 2019; Barbu et al., 2019; Djolonga et al., 2021; Funk et al., 2021; Gulrajani and Lopez-Paz, 2020; Hendrycks and Dietterich, 2019; Koh et al., 2021; Michaelis et al., 2019; Roy et al., 2018; Schölkopf et al., 2021). One instance of such models are agents that learn by interacting with a training environment and we would like them to generalize to other environments with different statistics (Ahmed et al., 2021; Cobbe et al., 2019; Ke et al., 2021; Pfister, Bauer, and Peters, 2019; Zhang et al., 2018). Consider the example of a robot with the task of moving a cube to a target position: Such an agent can easily fail as soon as some aspects of the environment differ from the training setup, e.g. the shape, color, and other object properties, or when transferring from simulation to real world.

Humans do not suffer from these pitfalls when transferring learned skills beyond a narrow training domain, presumably because they represent visual sensory data in a concise and useful manner (Anand et al., 2019; Gordon and Irwin, 1996; Lake et al., 2017; Marr, 1982; Spelke, 1990). Therefore, a particularly promising path is to base predictions and decisions on similar low-dimensional representations of our world (Barreto et al., 2017; Bengio, Courville, and Vincent, 2013; Dittadi, Drachmann, and Bolander, 2021; Finn et al., 2016; Kaiser et al., 2019; Stooke et al., 2021; Vinyals et al., 2019). The learned representation should facilitate efficient downstream learning (Anand et al., 2019; Eslami et al., 2018; Steenkiste et al., 2019; Stooke et al., 2021) and exhibit better generalization (Srinivas, Laskin, and Abbeel, 2020; Zhang et al., 2020). Learning such a representation from scratch for every downstream task and every new variation would be inefficient. If we learned to juggle three balls, we should be able to generalize to oranges or apples without learning again from scratch. We could even do it with cherimoyas, a fruit that we might have never seen before. We can effectively reuse our generic representation of the world.

We thus consider deep learning agents trained from pretrained representations and ask the following questions: To what extent do they generalize under distribution shifts similar to those mentioned above? Do they generalize in different ways or to different degrees depending on the type of distribution shift, including sim-to-real?

Can we predict the OOD generalization of downstream agents from properties of the pretrained representations?

To answer the questions above, we need our experimental setting to be realistic, diverse, and challenging, but also controlled enough for the conclusions to be sound. We therefore base our study on the robot platform introduced by Wüthrich et al. (2020). The scene comprises a robot finger with three joints that can be controlled to manipulate a cube in a bowl-shaped stage. Dittadi et al. (2021b) conveniently introduced a dataset of simulated and real-world images of this setup with ground-truth labels, which can be used to pretrain and evaluate representations. To train downstream agents, we adapted the simulated reinforcement learning benchmark CausalWorld from Ahmed et al. (2021) that was developed for this platform. Building upon these works, we design our experimental study as follows (see Fig. 6.1): First, we pretrain representations from static simulated images of the setup and evaluate a collection of representation metrics. Following prior work (Eslami et al., 2018; Ghadirzadeh et al., 2017; Ha and Schmidhuber, 2018; Nair et al., 2018; Van Hoof et al., 2016; Watter et al., 2015), we focus on autoencoder-based representations. Then, we train downstream agents from this fixed representation on a set of environments. Finally, we investigate the zero-shot generalization of these agents to new environments that are out of the training distribution, including the real robot.

The goal of this work is to provide the first systematic and extensive account of the OOD generalization of downstream RL agents in a robotic setup, and how this is affected by characteristics of the upstream pretrained representations. We summarize our contributions as follows:

- We train 240 representations and 11,520 downstream policies,[1] and systematically investigate their performance under a diverse range of distribution shifts.[2]
- We extensively analyze the relationship between the generalization of our RL agents and a substantial set of representation metrics.
- Notably, we find that a specific representation metric that measures the generalization of a simple downstream proxy task reliably predicts the generalization of downstream RL agents under the broad spectrum of OOD settings considered

[1] Training the representations required approximately 0.62 GPU years on NVIDIA Tesla V100. Training and evaluating the downstream policies required about 86.8 CPU years on Intel Platinum 8175M.

[2] Additional results and videos are provided at `https://sites.google.com/view/ood-rl`.

here. This metric can thus be used to select pretrained representations that will lead to more robust downstream policies.

- In the most challenging of our OOD scenarios, we deploy a subset of the trained policies to the corresponding real-world robotic platform, and observe surprising zero-shot sim-to-real generalization without any fine-tuning or domain randomization.

6.2 Background

In this section, we provide relevant background on the methods for representation learning and reinforcement learning, and on the robotic setup to evaluate out-of-distribution generalization.

Variational autoencoders. VAEs (Kingma and Welling, 2014; Rezende, Mohamed, and Wierstra, 2014) are a framework for optimizing a latent variable model $p_{\boldsymbol{\theta}}(\mathbf{x}) = \int_{\mathbf{z}} p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$ with parameters θ, typically with a fixed prior $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$, using amortized stochastic variational inference. A variational distribution $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ with parameters ϕ approximates the intractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. The approximate posterior and generative model, typically called encoder and decoder and parameterized by neural networks, are jointly optimized by maximizing a lower bound to the log likelihood (the ELBO):

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) \geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\right) = \mathcal{L}^{ELBO}_{\theta,\phi}(\mathbf{x}) \ . \tag{6.1}$$

In β-VAEs, the KL term is modulated by a factor β to enforce a more structured latent space (Burgess et al., 2018; Higgins et al., 2017a). While VAEs are typically trained without supervision, we also employ a form of weak supervision (Locatello et al., 2020b) that encourages disentanglement.

Reinforcement learning. A Reinforcement Learning (RL) problem is typically modeled as a Partially Observable Markov Decision Process (POMDP) defined as a tuple $(S, A, T, R, \Omega, O, \gamma, \rho_0, H)$ with states $s \in S$, actions $a \in A$ and observations $o \in \Omega$ determined by the state and action of the environment $O(o|s, a)$. $T(s_{t+1}|s_t, a_t)$ is the transition probability distribution function, $R(s_t, a_t)$ is the reward function, γ is the discount factor, $\rho_0(s)$ is the initial state distribution at the beginning of each episode, and H is the time horizon per episode. The objective in RL is to learn a

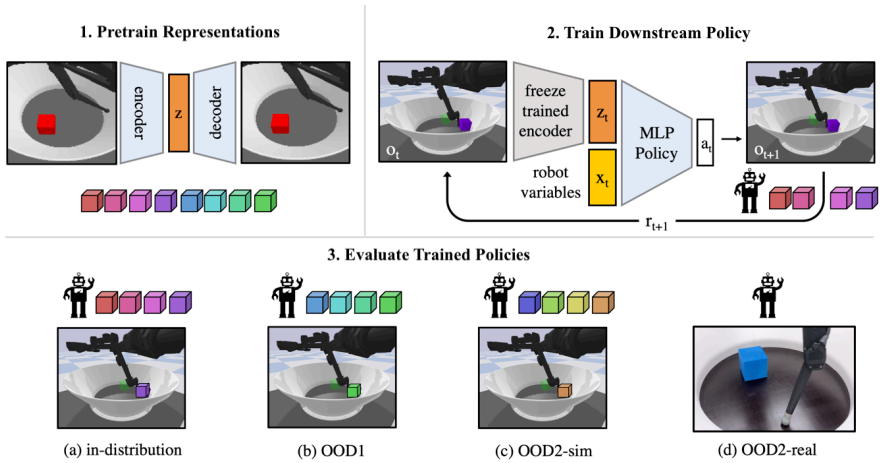


Figure 6.1. Overview of our experimental setup for investigating out-of-distribution generalization in downstream tasks. (1) We train 240 β-VAEs on the robotic dataset from Dittadi et al. (2021b). (2) We then train downstream policies to solve *object reaching* or *pushing*, using multiple random RL seeds per VAE. The input to a policy consists of the output of a pretrained encoder and additional task-related observable variables. Crucially, the policy is only trained on a subset of the cube colors from the pretraining dataset. (3) Finally, we evaluate these policies on their respective tasks in four different scenarios: (a) in-distribution, i.e. with cube colors used in policy training; (b) OOD1, i.e. with cube colours previously seen by the encoder but OOD for the policy; (c) OOD2-sim, having cube colours also OOD to the encoder; (d) sim-to-real zero-shot on the real-world setup.

policy $\pi : S \times A \to [0, 1]$, typically parameterized by a neural network, that maximizes the total discounted expected reward $J(\pi) = \mathbb{E}\big[\sum_{t=0}^{H} \gamma^t R(s_t, a_t)\big]$. There is a broad range of model-free learning algorithms to find π^* by policy gradient optimization or by learning value functions while trading off exploration and exploitation (Fujimoto, Hoof, and Meger, 2018; Haarnoja et al., 2018b; Schulman et al., 2015a; b; 2017; Silver et al., 2014; Sutton et al., 1999). Here, we optimize the objective above with *Soft Actor Critic* (SAC), an off-policy method that simultaneously maximizes the expected reward and the entropy $H(\pi(\cdot|s_t))$, and is widely used in control tasks due to its sample efficiency (Haarnoja et al., 2018b).

A robotic setup to evaluate out-of-distribution generalization. Our study is based on a real robot platform where a robotic finger with three joints manipulates a cube in a bowl-shaped stage (Wüthrich et al., 2020). We pretrain representations on a labeled dataset introduced by Dittadi et al. (2021b) which consists of simulated

and real-world images of this setup. This dataset has 7 underlying factors of variation (FoV): angles of the three joints, and position (x and y), orientation, and color of the cube. Some of these factors are correlated (Dittadi et al., 2021b), which may be problematic for representation learners, especially in the context of disentanglement (Chen et al., 2021; Träuble et al., 2021). After training the representations, we train downstream agents and evaluate their generalization on an adapted version of the simulated CausalWorld benchmark (Ahmed et al., 2021) that was developed for the same setup. Finally, we test sim-to-real generalization on the real robot.

Our experimental setup, illustrated in Fig. 6.1, allows us to systematically investigate a broad range of out-of-distribution scenarios in a controlled way. We pretrain our representations from this simulated dataset that covers 8 distinct cube colors. We then train an agent from this fixed representation on a subset of the cube colors, and evaluate it (1) on the same colors (this is the typical scenario in RL), (2) on the held-out cube colors that are still known to the encoder, or (3) OOD w.r.t. the encoder's training distribution, e.g. on novel colors and shapes or on the real world.

We closely follow the framework for measuring OOD generalization proposed by Dittadi et al. (2021b). In this framework, a representation is initially learned on a training set $\mathcal{D}$, and a simple downstream model is trained on a subset $\mathcal{D}_1 \subset \mathcal{D}$ to predict the ground-truth factors from the learned representation. Generalization is then evaluated by testing the downstream model on a set $\mathcal{D}_2$ that differs distributionally from $\mathcal{D}_1$, e.g. containing images corresponding to held-out values of a chosen factor of variation (FoV). Dittadi et al. (2021b) consider two flavors of OOD generalization depending on the choice of $\mathcal{D}_2$: First, the case when $\mathcal{D}_2 \subset \mathcal{D}$, i.e. the OOD test set is a subset of the dataset for representation learning. This is denoted by **OOD1** and corresponds to the scenario (2) from the previous paragraph. In the other scenario, referred to as **OOD2**, $\mathcal{D}$ and $\mathcal{D}_2$ are disjoint and distributionally different. This even stronger OOD shift corresponds to case (3) above. The generalization score for $\mathcal{D}_2$ is then measured by the (normalized) mean absolute prediction error across all FoVs except for the one that is OOD. Following Dittadi et al. (2021b), we use a simple 2-layer Multi-Layer Perceptron (MLP) for downstream factor prediction, we train one MLP for each FoV, and report the *negative* error. This simple and cheap generalization metric could serve as a convenient proxy for the generalization of more expensive downstream tasks. We refer to these generalization scores as GS-OOD1, GS-OOD2-sim, and GS-OOD2-real depending on the scenario.

The focus of Dittadi et al. (2021b) was to scale VAE-based approaches to more real-

istic scenarios and study the generalization of these simple downstream tasks, with a particular emphasis on disentanglement. Building upon their contributions, we can leverage the broader potential of this robotic setup with many more OOD2 scenarios to study our research questions: To what extent can agents generalize under distribution shift? Do they generalize in different ways depending on the type of shift (including sim-to-real)? Can we predict the OOD generalization of downstream agents from properties of the pretrained representations such as the GS metrics from Dittadi et al. (2021b)?

6.3 Study design

Robotic setup. Our setup is based on TriFinger (Wüthrich et al., 2020) and consists of a robotic finger with three joints that can be controlled to manipulate an object (e.g. a cube) in a bowl-shaped stage. The agent receives a camera observation consistent with the images in Dittadi et al. (2021b) and outputs a three-dimensional action. During training, which always happens in simulation, the agent only observes a cube of four possible colors, randomly sampled at every episode (see Fig. 6.1, step 2).

Distribution shifts. After training, we evaluate these agents in 7 environments: (1) the training environment, which is the typical setting in RL, (2) the OOD1 setting with cube colors that are OOD for the agent but still in-distribution for the encoder, (3) the more challenging OOD2-sim setting where the colors are also OOD for the encoder, (4-6) the OOD2 settings where the object colors are as in the 3 previous settings but the cube is replaced by a sphere (a previously unseen shape), (7) the OOD2-real setting, where we evaluate zero-shot sim-to-real transfer on the real robotic platform.

Tasks. We begin our study with the *object reaching* downstream control task, where the agent has to reach an object placed at an *arbitrary* random position in the arena. This is significantly more challenging than directly predicting the ground-truth factors, as the agent has to learn to reach the cube by acting on the joints, with a scalar reward as the only learning signal. Consequently, the compute required to learn this task is about 1,000 times greater than in the simple factor prediction case. We additionally include in our study a *pushing* task which consists of pushing an object to a goal position that is sampled at each episode. Learning this task takes one order of magnitude more compute than *object reaching*, likely due to the complex rigid-body

dynamics and object interactions. To the best of our knowledge, this is the most challenging manipulation task that is currently feasible on our setup. Ahmed et al. (2021) report solving a similar pushing task, but require the full ground-truth state to be observable.

Training the RL agents. The inputs at time t are the camera observation o_t and a vector of observable variables x_t containing the joint angles and velocities, as well as the target object position in *pushing*. We then feed the camera observation o_t into an encoder e that was pretrained on the dataset in Dittadi et al. (2021b). The result is concatenated with x_t, yielding a state vector $s_t = [x_t, e(o_t)]$. We then use SAC to train the policy with s_t as input. The policy, value, and Q networks are implemented as MLPs with 2 hidden layers of size 256. When training the policies, we keep the encoder frozen.

Model sweep. To shed light on the research questions outlined in the previous sections, we perform a large-scale study in which we train 240 representation models and 11,520 downstream policies, as described below. See Appendix B.1 for further implementation details.

- We train 120 β-VAEs (Higgins et al., 2017a) and 120 Ada-GVAEs (Locatello et al., 2020b) with a subset of the hyperparameter configurations and neural architecture from Dittadi et al. (2021b). Specifically, we consider $\beta \in \{1, 2, 4\}$, β annealing over $\{0, 50000\}$ steps, with and without input noise, and 10 random seeds per configuration. The latent space size is fixed to 10 following prior work (Chen et al., 2018; Kim and Mnih, 2018; Locatello et al., 2020b; Träuble et al., 2021).

- For *object reaching*, we train 20 downstream policies (varying random seed) for each of the 240 VAEs. The resulting 4,800 policies are trained for 400k steps (approximately 2,400 episodes).

- Since *pushing* takes substantially longer to train, we limit the number of policies trained on this task: We choose a subset of 96 VAEs corresponding to only 4 seeds, and then use 10 seeds per representation. The resulting 960 policies are trained for 3M steps (about 9,000 episodes).

- Finally, for both tasks we also investigate the role of regularization on the policy. More specifically, we repeat the two training sweeps from above (5,760 policies),

with the difference that now the policies are trained with L1 regularization on the first layer.

Limitations of our study. Although we aim to provide a sound and extensive empirical study, such studies are inevitably computationally demanding. Thus, we found it necessary to make certain design choices. For each of these choices, we attempted to follow common practice, in order to maintain our study as relevant, general, and useful as possible. One such decision is that of focusing on autoencoder-based representations. To answer our questions on the effect of upstream representations on the generalization of downstream policies, we need a diverse range of representations. How these representations are obtained is not directly relevant to answer our research question. Following Dittadi et al. (2021b), we chose to focus on β-VAE and Ada-GVAE models, as they were shown to provide a broad set of representations, including fully disentangled ones. Although we conjecture that other classes of representation learning algorithms should generally reveal similar trends as those found in our study, this is undoubtedly an interesting extension. As for the RL algorithm used in this work, SAC is known to be a particularly sample-efficient model-free RL method that is a popular choice in robotics (Haarnoja et al., 2018a; Kiran et al., 2021; Singh et al., 2019). Extensive results on pushing from ground-truth features on the same setup in Ahmed et al. (2021) indicate that methods like TD3 (Fujimoto, Hoof, and Meger, 2018) or PPO (Schulman et al., 2017) perform very similarly to SAC under the same reward structure and observation space. Thus, we expect the results of our study to hold beyond SAC. Another interesting direction is the study of additional regularization schemes on the policy network, an aspect that is often overlooked in RL. We expect the potential insights from extending the study along these axes to not justify the additional compute costs and corresponding carbon footprint. However, with improving efficiency and decreasing costs, we believe these could become worthwhile validation experiments in the future.

6.4 Results

We discuss our results in three parts: In Section 6.4.1, we present the training results of our large-scale sweep, and how policy regularization and different properties of the pretrained representations affect in-distribution reward. Section 6.4.2 gives an extensive account of which metrics of the pretrained representations predict OOD

generalization of the agents in simulated environments. Finally, in Section 6.4.3 we perform a similar evaluation on the real robot, in a zero-shot sim-to-real scenario.

6.4.1 Results in the training environment

Fig. 6.2 shows the training curves of all policies for *object reaching* and *pushing* in terms of the task-specific success metric. Here we use success metrics for interpretability, as their range is always $[0, 1]$. In *object reaching*, the success metric indicates progress from the initial end effector position to the optimal distance from the center of the cube. It is 0 if the final distance is not smaller than the initial distance, and 1 if the end effector is touching the center of a face of the cube. In *pushing*, the success metric is defined as the volumetric overlap of the cube with the goal cube, and the task can be visually considered solved with a score around 80%.

From the training curves we can conclude that both tasks can be consistently solved from pixels using pretrained representations. In particular, all policies on *object reaching* attain almost perfect scores. Unsurprisingly, the more complex *pushing* task requires significantly more training, and the variance across policies is larger. Nonetheless, almost all policies learn to solve the task satisfactorily.

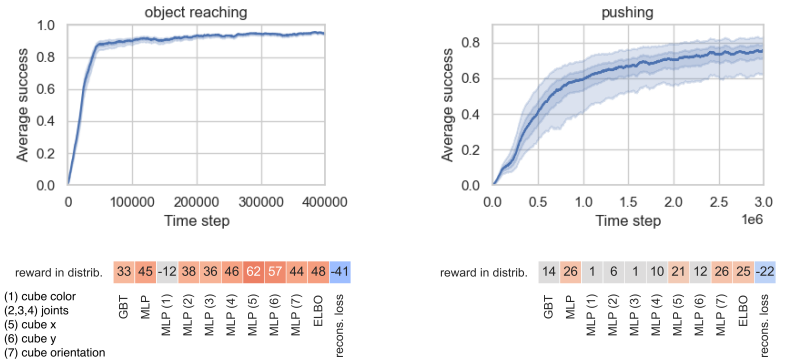


Figure 6.2. Top: Average training success, aggregated over *all* policies from the sweep (median, quartiles, 5th/95th percentiles). Bottom: Rank correlations between representation metrics and in-distribution reward (evaluated when the policies are fully trained), in the case without regularization. Correlations are color-coded in red (positive) or blue (negative) when statistically significant ($p<0.05$), otherwise they are gray.

To investigate the effect of representations on the training reward, we now compute its Spearman rank correlations with various supervised and unsupervised metrics of the representations (Fig. 6.2 bottom). By training reward here we mean the average reward of a fully trained policy over 200 episodes in the training environment (see Appendix B.1). On *object reaching*, the final reward correlates with the ELBO and the reconstruction loss. A simple supervised metric to evaluate a representation is how well a small downstream model can predict the ground-truth factors of variation. Following Dittadi et al. (2021b), we use the MLP10000 and GBT10000 metrics (simply MLP and GBT in the following), where MLPs and Gradient Boosted Trees (GBTs) are trained to predict the FoVs from 10,000 samples. The training reward correlates with these metrics as well, especially with the MLP accuracy. This is not entirely surprising: if an MLP can predict the FoVs from the representations, our policies using the same architecture could in principle retrieve the FoVs relevant for the task. Interestingly, the correlation with the overall MLP metric mostly stems from the cube pose FoVs, i.e. those that are not included in the ground-truth state x_t. These results suggest that these metrics can be used to select good representations for downstream RL. On the more challenging task of *pushing*, the correlations are milder but most of them are still statistically significant.

Summary. Both tasks can be consistently solved from pixels using pretrained representations. Unsupervised (ELBO, reconstruction loss) and supervised (ground-truth factor prediction) in-distribution metrics of the representations are correlated with reward in the training environment.

6.4.2 Out-of-distribution generalization in simulation

In- and out-of-distribution rewards. After training, the in-distribution reward correlates with OOD1 performance on both tasks (especially with regularization), but not with OOD2 performance (see Fig. 6.3). Moreover, rewards in OOD1 and OOD2 environments are moderately correlated across tasks and regularization settings.

Unsupervised metrics and informativeness. In Fig. 6.4 (left) we assess the relation between OOD reward and in-distribution metrics (ELBO, reconstruction loss, MLP, and GBT). Both ELBO and reconstruction loss exhibit a correlation with OOD1 reward, but not with OOD2 reward. These unsupervised metrics can thus be useful for selecting representations that will lead to more robust downstream RL tasks,

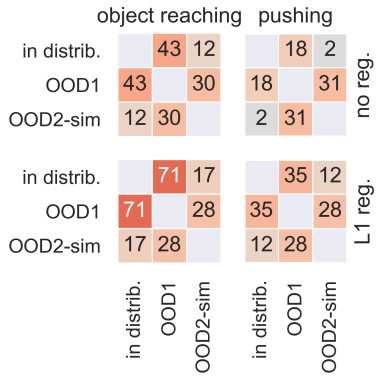


Figure 6.3. Correlations between training (in distrib.) and OOD rewards ($p<0.05$).

as long as the encoder is in-distribution. While the GBT score is not correlated with reward under distribution shift, we observe a significant correlation between OOD1 reward and the MLP score, which measures downstream factor prediction accuracy of an MLP with the same architecture as the one parameterizing the policies. As in Section 6.4.1, we further investigate the source of this correlation, and find it in the pose parameters of the cube. Correlations in the OOD2 setting are much weaker, thus we conclude that these metrics do not appear helpful for model selection in this case. Our results on *pushing* confirm these conclusions although correlations are generally weaker, presumably due to the more complicated nature of this task. An extensive discussion is provided in Appendix B.2.2.

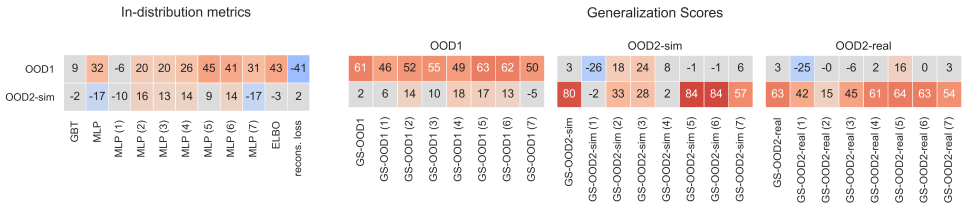


Figure 6.4. Rank correlations of representation properties with OOD1 and OOD2 reward on *object reaching* without regularization. Numbering when splitting metrics by FoV: (1) cube color; (2–4) joint angles; (5–7) cube position and rotation. Correlations are color-coded as described in Fig. 6.2.

Correlations with generalization scores. Here we analyze the link between generalization in RL and the generalization scores (GS) discussed in Section 6.2, which measure the generalization of downstream FoV predictors *out of distribution*, as opposed to the MLP and GBT metrics considered above. For both OOD scenarios, the distribution shifts underlying these GS scores are the same as the ones in the RL tasks in simulation. We summarize our findings in Fig. 6.4 (right) on the *object reaching* task. Reward in the OOD1 setting is significantly correlated with the GS-OOD1 metric of the pretrained representation. We observe an even stronger correlation between the reward in the simulated OOD2 setting and the corresponding GS-OOD2-sim and GS-OOD2-real scores. On a per-factor level, we see that the source of the observed correlations primarily stems from the generalization scores w.r.t. the pose parameters of the cube. The OOD generalization metrics can therefore be used as proxies for the corresponding form of generalization in downstream RL tasks. This has practical implications for the training of RL downstream policies which are generally known to be brittle to distribution shifts, as we can measure a representation's generalization score from a few labeled images. This allows for selecting representations that yield more robust downstream policies.

Disentangled representations. Disentanglement has been shown to be helpful for downstream performance and OOD1 generalization even with MLPs (Dittadi et al., 2021b). However, in *object reaching*, we only observe a weak correlation with some disentanglement metrics (Fig. 6.5). In agreement with (Dittadi et al., 2021b), disentanglement does not correlate with OOD2 generalization. The same study observed that disentanglement correlates with the informativeness of a representation. To understand if these weak correlations originate from this common confounder, we investigate whether they persist after adjusting for MLP FoV prediction accuracy. Given two representations with similar MLP accuracy, does the more disentangled one exhibit better OOD1 generalization? To measure this we predict success from the MLP accuracy using kNN (k=5) (Locatello et al., 2019a) and compute the residual reward by subtracting the amount of reward explained by the MLP metric. Fig. 6.5 shows that this resolves the remaining correlations with disentanglement. Thus, for the RL downstream tasks considered here, disentanglement per se does not seem to be useful for OOD generalization. We present similar results on *pushing* in Appendix B.2.2.

Policy regularization and observation noise. It might seem unsurprising that disentanglement is not useful for generalization in RL, as MLP policies do not have any explicit inductive bias to exploit it. Thus, we attempt to introduce such induc-

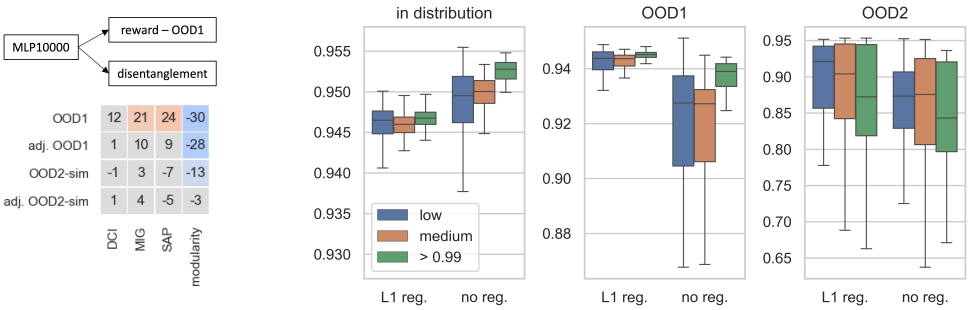


Figure 6.5. Box plots: fractional success on *object reaching* split according to low (blue), medium-high (orange), and almost perfect (green) disentanglement. L1 regularization in the first layer of the MLP policy has a positive effect on OOD1 and OOD2 generalization with minimal sacrifice in terms of training reward (see scale). **Correlation matrix** (left): although we observe a mild correlation between some disentanglement metrics and OOD1 (but not OOD2) generalization, this does not hold when adjusting for representation informativeness. Correlations are color-coded as described in Fig. 6.2. We use disentanglement metrics from Chen et al. (2018), Eastwood and Williams (2018), Kumar, Sattigeri, and Balakrishnan (2018), and Ridgeway and Mozer (2018).

tive bias by repeating all experiments with L1 regularization on the first layer of the policy. Although regularization improves OOD1 and OOD2 generalization in general (see box plots in Fig. 6.5), we observe no clear link with disentanglement. Furthermore, in accordance with Dittadi et al. (2021b), we find that observation noise when training representations is beneficial for OOD2 generalization. See Appendix B.2.2 for a detailed discussion.

Stronger OOD shifts: evaluating on a novel shape. On *object reaching*, we also test generalization w.r.t. a novel shape by replacing the cube with a sphere. This corresponds to a strong OOD2-type shift, since shape was never varied when training the representations. Surprisingly, the policies appear to be robust to the novel shape. In fact, when the sphere has the same colors that the cube had during policy training, *all* policies get closer than 5 cm to the sphere on average, with a mean success metric of 95%. On sphere colors from the OOD1 split, more than 98.5% move the finger closer than this threshold, and on the strongest distribution shift (OOD2-sim colors, and cube replaced by sphere) almost 70% surpass that threshold with an average success metric above 80%.

Summary. (1) In- and out-of-distribution rewards are correlated, as long as the representation remains in its training distribution (OOD1). (2) Similarly, in-distribution representation metrics (both unsupervised and supervised) predict OOD1 reward, but are not reliable when the representation is OOD (OOD2). (3) Disentanglement does not correlate with generalization in our experiments, while (4) input noise when training representations is beneficial for OOD2 generalization. (5) Most notably, the GS metrics, which measure generalization under distribution shifts, are significantly correlated with RL performance under similar distribution shifts. We thus recommend using these convenient proxy metrics for selecting representations that will yield robust downstream policies.

6.4.3 Deploying policies to the real world

We now evaluate a large subset of the agents on the real robot without fine-tuning, quantify their zero-shot sim-to-real generalization, and find metrics that correlate with real-world performance.

Reaching. We choose 960 policies trained in simulation, based on 96 representations and 10 random seeds, and evaluate them on two (randomly chosen, but far apart) goal positions using a red cube. While a red cube was in the training distribution, we consider this to be OOD2 because real-world images represent a strong distribution shift for the encoder (Dittadi et al., 2021b; Djolonga et al., 2021). Although sim-to-real in robotics is considered to be very challenging without domain randomization

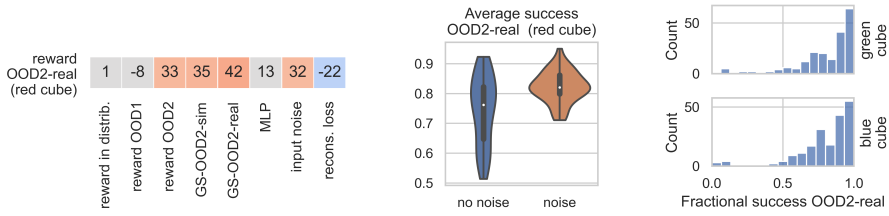


Figure 6.6. Zero-shot sim-to-real on *object reaching* on over 2,000 episodes. **Left:** Rank-correlations on the real platform with a red cube (color-coded as described in Fig. 6.2). **Middle**: Training encoders with additive noise improves sim-to-real generalization. **Right**: Histogram of fractional success in the more challenging OOD2-real-{green,blue} scenario from 50 policies across 4 different goal positions.

or fine-tuning (Finn et al., 2017; Rusu et al., 2017; Tobin et al., 2017), many of our policies obtain a high fractional success without resorting to these methods. In addition, in Fig. 6.6 (left) we observe significant correlations between zero-shot real-world performance and some of the previously discussed metrics. First, there is a positive correlation with the OOD2-sim reward: Policies that generalize to unseen cube colors in simulation also generalize to the real world. Second, representations with high GS-OOD2-sim and (especially) GS-OOD2-real scores are promising candidates for sim-to-real transfer. Third, if no labels are available, the weaker correlation with the reconstruction loss on the simulated images can be exploited for representation selection. Finally, as observed by Dittadi et al. (2021b) for simple downstream tasks, input noise while learning representations is beneficial for sim-to-real generalization (Fig. 6.6, middle).

Based on these findings, we select 50 policies with a high GS-OOD2-real score, and evaluate them on the real world with a green and a blue cube, which is an even stronger OOD2 distribution shift. In Fig. 6.6 (right), where metrics are averaged over 4 cube positions per policy, we observe that most policies can still solve the task: approximately 80% of them position the finger less than 5 cm from the cube. Lastly, we repeat the evaluations on the green sphere that we previously performed in simulation, and observe that many policies successfully reach this completely novel object. See Appendix B.2.3 and the project website for additional results and videos of deployed policies.

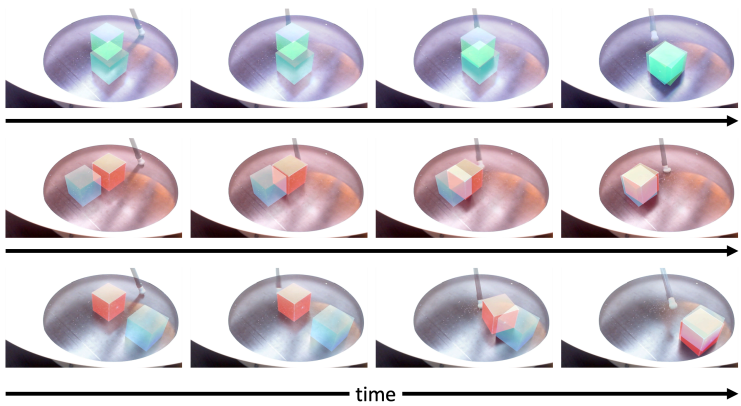


Figure 6.7. We select pushing policies with high GS-OOD2-real score. When deployed on the real robot without fine-tuning, they succeed in pushing the cube to a specified goal position (transparent blue cube).

Pushing. We now test whether our real-world findings on *object reaching* also hold for *pushing*. We again select policies with a high GS-OOD2-real score and encoders trained with input noise. We record episodes on diverse goal positions and cube colors to support our finding that pushing policies in simulation can generalize to the real robot. In Fig. 6.7, we show three representative episodes with successful task completions and refer to the project site for video recordings and further episodes.

Summary. Policies trained in simulation can solve the task on the real robot without domain randomization or fine-tuning. Reconstruction loss, encoder robustness, and OOD2 reward in simulation are all good predictors of real-world performance. For real-world applications, we recommend using GS-OOD2-sim or GS-OOD2-real for model selection, and training the encoder with noise.

6.5 Other related work

A key unsolved challenge in RL is the brittleness of agents to distribution shifts in the environment, even if the underlying structure is largely unchanged (Ahmed et al., 2021; Cobbe et al., 2019). This is related to studies on representation learning and generalization in downstream tasks (Chaabouni et al., 2020; Dittadi et al., 2022a; Esmaeili et al., 2019a; Gondal et al., 2019; Steenbrugge et al., 2018), as well as domain generalization (see Wang et al. (2021) for an overview). More specifically for RL, Higgins et al. (2017b) focus on domain adaptation and zero-shot transfer in DeepMind Lab and MuJoCo environments, and claim disentanglement improves robustness. To obtain better transfer capabilities, Asadi, Abel, and Littman (2020) argue for discretizing the state space in continuous control domains by clustering states where the optimal policy is similar. Kulkarni et al. (2015) propose geometric object representations by means of keypoints or image-space coordinates and Wulfmeier et al. (2021) investigate the effect of different representations on the learning and exploration of different robotics tasks. Transfer becomes especially challenging from the simulation to the real world, a phenomenon often referred to as the sim-to-real gap. This is particularly crucial in RL, as real-world training is expensive, requires sample-efficient methods, and is sometimes unfeasible if the reward structure requires accurate ground truth labels (Dulac-Arnold, Mankowitz, and Hester, 2019; Kormushev, Calinon, and Caldwell, 2013). This issue is typically tackled with large-scale domain randomization in simulation (Akkaya et al., 2019; James et al., 2019).

6.6 Conclusion

Robust out-of-distribution (OOD) generalization is still one of the key open challenges in machine learning. We attempted to answer central questions on the generalization of reinforcement learning agents in a robotics context, and how this is affected by pretrained representations. We presented a large-scale empirical study in which we trained over 10,000 downstream agents given pretrained representations, and extensively tested them under a variety of distribution shifts, including sim-to-real. We observed agents that generalize OOD, and found that some properties of the pretrained representations can be useful to predict which agents will generalize better. We believe this work brings us one step closer to understanding the generalization abilities of learning systems, and we hope that it encourages many further important studies in this direction.

Acknowledgements

We would like to thank Anirudh Goyal, Georg Martius, Nasim Rahaman, Vaibhav Agrawal, Max Horn, and the Causality group at the MPI for useful discussions and feedback. We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting FT. Part of the experiments were generously supported with compute credits by Amazon Web Services.

CHAPTER 7

Paper III: Generalization and Robustness Implications in Object-Centric Learning

Authors: Andrea Dittadi, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, Francesco Locatello.

Status: Published in *International Conference on Machine Learning*, 2022.

Abstract. The idea behind object-centric representation learning is that natural scenes can better be modeled as compositions of objects and their relations as opposed to distributed representations. This inductive bias can be injected into neural networks to potentially improve systematic generalization and performance of downstream tasks in scenes with multiple objects. In this paper, we train state-of-the-art unsupervised models on five common multi-object datasets and evaluate segmentation metrics and downstream object property prediction. In addition, we study generalization and robustness by investigating the settings where either a single object is out of distribution—e.g., having an unseen color, texture, or shape—or global properties of the scene are altered—e.g., by occlusions, cropping, or increasing the number of objects. From our experimental study, we find object-centric representations to be useful for downstream tasks and generally robust to most distribution shifts affecting objects. However, when the distribution shift affects the input in a less structured manner, robustness in terms of segmentation and downstream task performance may vary significantly across models and distribution shifts.

7.1 Introduction

In object-centric representation learning, we make the assumption that visual scenes are composed of multiple entities or objects that interact with each other, and exploit this compositional property as inductive bias for neural networks. Informally, the goal is to find transformations r of the data $\mathbf{x}$ into a *set* of vector representations $r(\mathbf{x}) = \{\mathbf{z}_k\}$ each corresponding to an individual object, without supervision (Burgess et al., 2019; Chen, Deng, and Ahn, 2020; Crawford and Pineau, 2019; Engelcke et al., 2020; Eslami et al., 2016; Greff et al., 2019; Greff, Steenkiste, and Schmidhuber, 2017; Gregor et al., 2015; Kosiorek et al., 2018; Lin et al., 2020b; Locatello et al., 2020d; Mnih, Heess, Graves, et al., 2014; Weis et al., 2020; Yuan, Li, and Xue, 2019). Relying on this inductive bias, object-centric representations are conjectured to be more robust than distributed representations, and to enable the systematic generalization typical of symbolic systems while retaining the expressiveness of connectionist approaches (Bengio, Courville, and Vincent, 2013; Greff, Steenkiste, and Schmidhuber, 2020; Lake et al., 2017; Schölkopf et al., 2021). Grounding for these claims comes mostly from cognitive psychology and neuroscience (Spelke, 1990; Téglás et al., 2011; Wagemans, 2015). E.g., infants learn about the physical properties of objects as entities that behave consistently over time (Baillargeon, Spelke, and Wasserman, 1985; Spelke and Kinzler, 2007) and are able to re-apply their knowledge to new scenarios involving previously unseen objects (Dehaene, 2020). Similarly, in complex machine learning tasks like physical modelling and reinforcement learning, it is common to train from the internal representation of a simulator (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020) or of a game engine (Berner et al., 2019; Vinyals et al., 2019) rather than from raw pixels, as more abstract representations facilitate learning. Finally, learning to represent objects separately is a crucial step towards learning causal models of the data from high-dimensional observations, as objects can be interpreted as causal variables that can be manipulated independently (Schölkopf et al., 2021). Such causal models are believed to be crucial for human-level generalization (Pearl, 2009; Peters, Janzing, and Schölkopf, 2017), but traditional causality research assumes causal variables to be given rather than learned (Schölkopf, 2019).

As object-centric learning developed recently as a subfield of representation learning, we identify three key hypotheses and design systematic experiments to test them. (1) *The unsupervised learning of objects as pretraining task is useful for downstream tasks.* Besides learning to separate objects without supervision, current approaches are expected to separately represent information about each object's properties, so

that the representations can be useful for arbitrary downstream tasks. (2) *In object-centric models, distribution shifts affecting a single object do not affect the representations of other objects.* If objects are to be represented independently of each other to act as compositional building blocks for higher-level cognition (Greff, Steenkiste, and Schmidhuber, 2020), changes to one object in the input should not affect the representation of the unchanged objects. This should hold even if the change leads to an object being out of distribution (OOD). (3) *Object-centric models are generally robust to distribution shifts, even if they affect global properties of the scene.* Even if the whole scene is OOD—e.g., if it contains more objects than in the training set—object-centric approaches should be robust thanks to their inductive bias.

In this paper, we systematically investigate these three concrete hypotheses by re-implementing popular unsupervised object discovery approaches and testing them on five multi-object datasets.[1] We find that: (1) Object-centric models achieve good downstream performance on property prediction tasks. We also observe a strong correlation between segmentation metrics, reconstruction error, and downstream property prediction performance, suggesting potential model selection strategies. (2) If a single object is out of distribution, the overall segmentation performance is not strongly impacted. Remarkably, the downstream prediction of in-distribution (ID) objects is mostly unaffected. (3) Under more global distribution shifts, the ability to separate objects depends significantly on the model and shift at hand, and downstream performance may be severely affected.

As an additional contribution, we provide a library[2] for benchmarking object-centric representation learning, which can be extended with more datasets, methods, and evaluation tasks. We hope this will foster further progress in the learning and evaluation of object-centric representations.

7.2 Study design and hypotheses

Problem definition: Vanilla deep learning architectures learn distributed representations that do not capture the compositional properties of natural scenes—see, e.g., the "superposition catastrophe" (Bowers et al., 2014; Greff, Steenkiste, and Schmid-

[1] Training and evaluating all the models for the main study requires approximately 1.44 GPU years on NVIDIA V100.

[2] https://github.com/addtt/object-centric-library

huber, 2020; Von Der Malsburg, 1986). Even in disentangled representation learning (Chen et al., 2018; Eastwood and Williams, 2018; Higgins et al., 2017a; Kim and Mnih, 2018; Kumar, Sattigeri, and Balakrishnan, 2018; Ridgeway and Mozer, 2018), factors of variations are encoded in a vector representation that is the output of a standard CNN encoder. This introduces an unnatural ordering of the objects in the scene and fails to capture its compositional structure in terms of objects. Formally defining objects is challenging (Greff, Steenkiste, and Schmidhuber, 2020) and there is no consensus even outside of machine learning (Green, 2019; Smith, 1998). Greff, Steenkiste, and Schmidhuber (2020) put forth three properties for object-centric representations: *separation*, i.e., object features in the set of vectors $r(\mathbf{x})$ do not interact with each other, and each object is individually captured in a single element of $r(\mathbf{x})$; *common format*, i.e., each element of $r(\mathbf{x})$ shares the same representational format; and *disentanglement*, i.e., each element of $r(\mathbf{x})$ is represented in a disentangled format that exposes the factors of variation. In this paper, we consider representations $r(\mathbf{x})$ that are sets of vectors with each element sharing the representational format. We take a pragmatic perspective and focus on two clear desiderata for object-centric approaches:

Desideratum 1: *Object embodiment.* The representation should contain information about the object's location and its embodiment in the scene. As we focus on unsupervised object discovery, this translates to segmentation masks. This is related to separation and common format, as the decoder is applied to the elements of $r(\mathbf{x})$ with shared parameters.

Desideratum 2: *Informativeness of the representation.* Instead of learning disentangled representations of objects, which is challenging even in single-object scenarios (Locatello et al., 2019b), we want the representation to contain useful information for downstream tasks, not necessarily in a disentangled format. We define objects through their properties as annotated in the datasets we consider, and predict these properties from the representations. Note that this may not be the only way to define objects (e.g., defining faces and edges as objects and deducing shapes as composition of those). The fact that existing models learn informative representations is our first hypothesis (see below).

Design principle: These desiderata offer well-defined quantitative evaluations for object-centric approaches and we want to understand the implications of learning such representations. To this end, we train four different state-of-the-art methods on five datasets, taking hyperparameter configurations from the respective publications and adapting them to improve performance when necessary. Assuming these models

succeeded in learning an object-centric representation, we investigate the following hypotheses.

Hypothesis 1: *The unsupervised learning of objects as pretraining task is useful for downstream tasks.* Existing empirical evaluations largely focus on our *Desideratum 1* and evaluate the performance at test time in terms of segmentation metrics. The hope, however, is that the representation would be useful for other downstream tasks besides segmentation (*Desideratum 2*). We test this hypothesis by training small downstream prediction models on the frozen object-representations with shared parameters to predict the object properties. We match the predictions to the ground-truth properties with the Hungarian algorithm (Kuhn, 1955) following Locatello et al. (2020d).

Hypothesis 2: *In object-centric models, distribution shifts affecting a single object do not affect the representations of other objects.* A change in the properties of one object in the input should not affect the representation of the other objects. Even OOD objects with previously unseen properties should be segmented correctly by a network that learned the notion of objects (Greff, Steenkiste, and Schmidhuber, 2020; Schölkopf et al., 2021). We test this hypothesis by (1) evaluating the segmentation of the scene after the distribution shift, and (2) training downstream models to predict object properties, and evaluating them on representations extracted from scenes with one OOD object. More specifically, we test changes in the shape, color, or texture of one object.

Hypothesis 3: *Object-centric models are generally robust to distribution shifts, even if they affect global properties of the scene.* Early evidence (Romijnders et al., 2021) points to the conjecture that learning object-centric representations biases the network towards learning more robust representations of the overall scene. Intuitively, the notion of objects is an additional inductive bias for the network to exploit to maintain accurate predictions if simple global properties of the scene are altered. We test this hypothesis by training downstream models to predict object properties, and evaluating them on representations of scenes with OOD global properties. In this case, we test robustness by cropping, introducing occlusions, and increasing the number of objects.

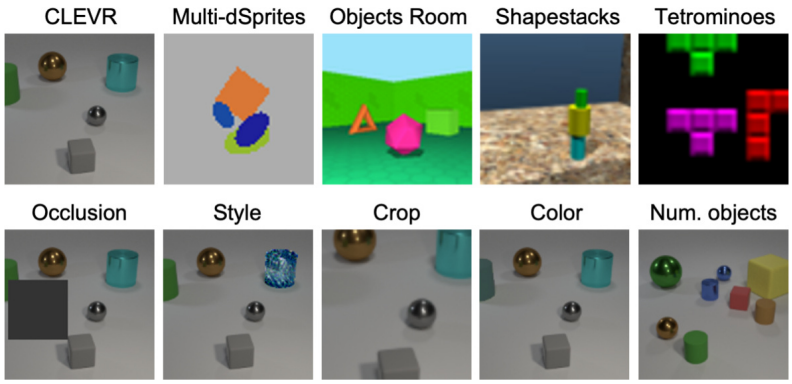


Figure 7.1. Top: examples from the five datasets in this study. **Bottom:** distribution shifts in CLEVR.

7.3 Experimental setup

Here we provide an overview of our experimental setup. After introducing the relevant models and datasets, we outline the evaluation protocols for segmentation accuracy (Desideratum 1) and downstream task performance (Desideratum 2). Then, we discuss the distribution shifts that we use to test robustness—the aforementioned evaluations are repeated once again under these distribution shifts. We conclude with a discussion on the limitations of this study.

Models and datasets. We implement four state-of-the-art object-centric models—MONet (Burgess et al., 2019), GENESIS (Engelcke et al., 2020), Slot Attention (Locatello et al., 2020d), and SPACE (Lin et al., 2020b)—as well as vanilla variational autoencoders (VAEs) (Kingma and Welling, 2014; Rezende, Mohamed, and Wierstra, 2014) as baselines for distributed representations. We use one VAE variant with a broadcast decoder (Watters et al., 2019) and one with a regular convolutional decoder. See Appendix C.1 for an overview of the models with implementation details. We then collect five popular multi-object datasets: *Multi-dSprites*, *Objects Room*, and *Tetrominoes* from DeepMind's Multi-Object Datasets collection (Kabra et al., 2019), *CLEVR* (Johnson et al., 2017), and *Shapestacks* (Groth et al., 2018). The datasets are shown in Fig. 7.1 (top row) and described in detail in Appendix C.2. For each dataset, we define train, validation, and test splits. The test splits, which always contain at least 2000 images, are exclusively used for evaluation. We train each model on all datasets, using 10 random seeds for object-centric models and 5 for each VAE

variant, resulting in 250 models in total.

Metrics. We evaluate the segmentation accuracy of object-centric models with the Adjusted Rand Index (ARI) (Hubert and Arabi, 1985), Segmentation Covering (SC) (Arbelaez et al., 2010), and mean Segmentation Covering (mSC) (Engelcke et al., 2020). For all models, we additionally evaluate reconstruction quality via the mean squared error (MSE). Appendix C.3.1 includes detailed definitions of these metrics.

Downstream property prediction. We evaluate object-centric representations by training downstream models to predict ground-truth object properties from the representations. More specifically, exploiting the fact that object slots share a common representational format, a single downstream model f can be used to predict the properties of each object independently: for each slot representation $\mathbf{z}_k$ we predict a vector of object properties $\hat{\mathbf{y}}_k = f(\mathbf{z}_k)$. As in previous work on object property prediction (Locatello et al., 2020d), each model simultaneously predicts all properties of an object. For learning, we use the cross-entropy loss for categorical properties and MSE for numerical properties, and denote by $\ell(\hat{\mathbf{y}}_k, \mathbf{y}_m)$ the overall loss for a single object, where $\mathbf{y}_m$ are its ground-truth properties. Here $k \in \{1, \dots, K\}$ and $m \in \{1, \dots, M\}$ with K the number of slots and M the number of objects. In order to optimize the downstream models, the vector $\hat{\mathbf{y}}_k$ (the properties predicted from the kth representational slot) needs to be matched to the ground-truth properties $\mathbf{y}_m$ of the mth object. This is done by computing a $M \times K$ matrix of *matching losses* for each slot–object pair, and then solving the assignment problem using the Hungarian algorithm (Kuhn, 1955) to minimize the total matching loss, which is the sum of $\min(M, K)$ terms from the loss matrix. As matching loss we use either the negative cosine similarity between predicted and ground-truth masks (as in Greff et al. (2019)), or the downstream loss $\ell(\hat{\mathbf{y}}_k, \mathbf{y}_m)$ itself (as in Locatello et al. (2020d)). In the following, we will refer to these strategies as *mask matching* and *loss matching*, respectively. For property prediction, we use 4 different downstream models: a linear model, and MLPs with up to 3 hidden layers of size 256 each. Given a pretrained object-centric model, we train each downstream model on the representations of 10 000 images. The downstream models are then tested on 2000 held-out images from the test set, which may exhibit distribution shifts as discussed below. Further details on this evaluation are provided in Appendix C.3.2

Evaluating distributed representations. Since in non-slot-based models, such as classical VAEs, the representations of the single objects are not readily available, matching representations to objects for downstream property prediction is not triv-

ial. Although this is an inherent limitation of distributed representations, we are nevertheless interested in evaluating their usefulness. Using the matching framework presented above, we require the downstream model f to output the predicted properties of *all* objects, and then match these with the true object properties to evaluate prediction quality. Our downstream model in this case will thus take as input the entire representation $\mathbf{z} = r(\mathbf{x})$ (which is now a single vector rather than a set of vectors) and output the predictions for all objects together as a vector $f(\mathbf{z})$. Finally, we split $f(\mathbf{z})$ into K vectors $\{\hat{\mathbf{y}}_k\}_{k=1}^K$, where K loosely corresponds to the number of slots in object-centric models. At this point, we can compute the loss $\ell(\hat{\mathbf{y}}_k, \mathbf{y}_m)$ for each pair, as usual. We now consider two matching strategies: As before, *loss matching* simply defines the matching loss of a slot–object pair as the prediction loss itself. In the *deterministic matching* strategy, following Greff et al. (2019), we lexicographically sort objects according to a canonical order of object properties. Calling π the permutation that defines this sorting, the kth slot is deterministically matched with the mth object, where $m = \pi^{-1}(k)$.

Baselines. To correctly assess performance on downstream tasks, it is fundamental to compare with sensible baselines. Here we consider as baseline the best performance that can be achieved by a downstream model that outputs a constant vector that does not depend on the image. When predicting properties independently for each object (in slot-based models), the optimal solution is to predict the mean of continuous properties and the majority class for categorical ones. When using deterministic matching in the distributed case, the downstream model can exploit the predefined total order to predict more accurately than random guessing even without using information from the input (this effect is non-negligible only for the properties that are most significant in the order). Finally, in a few cases, loss matching for distributed representations can be significantly better than deterministic matching.[3] For simplicity, for both matching strategies in the distributed case, we directly learn a vector $\hat{\mathbf{y}}$ by gradient descent to minimize the prediction loss. As this depends on random initialization and optimization dynamics, we repeat this for 10 random seeds and report error bars in the plots.

Distribution shifts. We test the robustness of the learned representations under two classes of distribution shifts: one where one object goes OOD, and one where global properties of the scene are changed. All such distribution shifts occur at test time,

[3]Intuitively, a (constant) diverse set of uninformed predictions $\{\hat{\mathbf{y}}_k\}$ might be sufficient for the matching algorithm to find suitable enough objects for most predictions.

i.e., the unsupervised models are always trained on the original datasets. To evaluate generalization to distribution shifts affecting a *single object*, we systematically induce changes in the color, shape, and texture of objects. To change color, we apply a random color shift to one random object in the scene, using the available masks (we do not do this in Multi-dSprites, as the training distribution covers the entire RGB color space). To test robustness to unseen textures, we apply neural style transfer (Gatys, Ecker, and Bethge, 2016) to one random object in each scene, using *The Great Wave off Kanagawa* as style image. When either a new color or a new texture is introduced, prediction of material (in CLEVR only) and color is not performed. To introduce a new shape, we select images from Multi-dSprites that have at most 4 objects (in general, they have up to 5), and add a randomly colored triangle, in a random position, at a random depth in the object stack. In this case, shape prediction does not apply. Finally, to test robustness to *global* changes in the scene, we change the number of objects (in CLEVR only), introduce occlusions (a gray square at a random location), or crop images at the center and restore their original size via bilinear interpolation. See Fig. 7.1 for examples, and Appendix C.3.3 for further details.

Limitations of this study. While we aim to conduct a sound and informative experimental study to answer the research questions from Section 7.3, inevitably there are limitations regarding datasets, models, and evaluations. Although the datasets considered here vary significantly in complexity and visual properties, they all consist of synthetic images where object properties are independent of each other and independent between objects. Regarding object-centric models, we only focus on autoencoder-based approaches that model a scene as a mixture of components. As official implementations are not always available, and none of the methods in this work has been applied to all the datasets considered here, we re-implement these methods and choose hyperparameters following a best-effort approach. Finally, we only consider the downstream task of object property prediction, and assess generalization using only a few representative single-object and global distribution shifts.

7.4 Results

In this section, we highlight our findings with plots that are representative of our main results. The full experimental results are presented in Appendix C.4. In Section 7.4.1 we focus on the different evaluation metrics and the performance we obtained retraining the methods considered in this study. We then focus on our three hypotheses

in Sections 7.4.2 to 7.4.4.

7.4.1 Learning and evaluating object discovery

Since all methods included in our study were originally evaluated only on a subset of the datasets and metrics considered here, we first test how well these models perform.

Fig. 7.2 shows the segmentation performance of the models in terms of ARI across models, datasets, and random seeds. Fig. C.3 in Appendix C.4 provides an overview of the reconstruction MSE and all segmentation metrics. Although these results are in line with published work, we observe substantial differences in the ranking between models depending on the metric. This indicates that, in practice, these metrics are not equivalent for measuring object discovery.

This is confirmed in Fig. 7.3, which shows rank correlations between metrics on different datasets (aggregating over different models). We also observe a strong negative correlation between ARI and MSE across models and datasets, suggesting that models that learn to more accurately reconstruct the input tend to better segment objects according to the ARI score. This trend is less consistent for the other segmentation metrics, as MSE significantly correlates with mSC in only three datasets (Multi-dSprites, Objects Room, and CLEVR), and with SC in two (Multi-dSprites and Objects Room). SC and mSC measure very similar segmentation notions and therefore are significantly correlated in all datasets, although to a varying extent. However, they correlate with ARI only on two and three datasets, respectively (the same datasets where they correlate with the MSE).

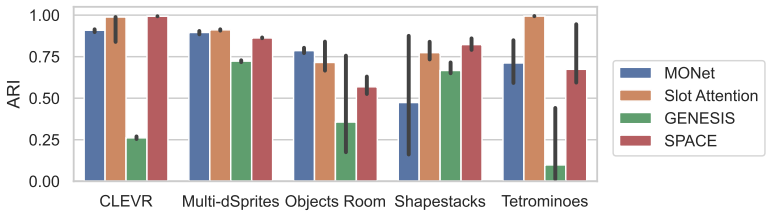


Figure 7.2. ARI of all models and datasets on 2000 test images. Medians and 95% confidence intervals with 10 seeds.

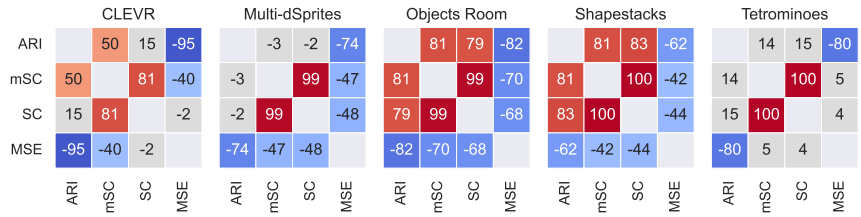


Figure 7.3. Spearman rank correlations between evaluation metrics across models and random seeds (color-coded only when p<0.05).

Summary: We observe strong differences in performance and ranking between the models depending on the evaluation metric. In the tested datasets, we find that the ARI, which requires ground-truth segmentation masks to compute, correlates particularly well with the MSE, which is unsupervised and provides training signal.

7.4.2 Usefulness for downstream tasks (Hypothesis 1)

To test Hypothesis 1, we first evaluate whether frozen object-centric representations can be used to train downstream models measuring Desideratum 2 from Section 7.2. As discussed in Section 7.3, this type of downstream task requires matching the true object properties with the predictions of the downstream model. In the following, we will only present results obtained with *loss matching*, and show results for other matching strategies in Appendix C.4.

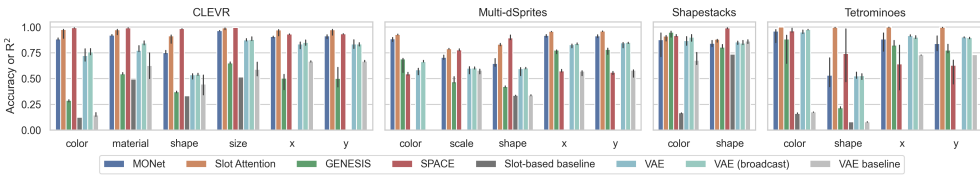


Figure 7.4. Comparison of downstream property prediction performance for object-centric (slot-based) and distributed (VAE) representations, using an MLP with one hidden layer as downstream model. The metric is accuracy for categorical properties or R^2 for numerical ones. The baselines in gray indicate the best performance that can be achieved by a model that outputs a constant vector that does not depend on the input. The bars show medians and 95% confidence intervals with 10 random seeds.

Fig. 7.4 shows downstream prediction performance on all datasets and models, when the downstream model is a single-layer MLP. Although results vary across datasets and models, accurate prediction of object properties seems to be possible in most of the scenarios considered here. Fig. C.5 in Appendix C.4 shows similar results when using a linear model or MLPs with up to 3 hidden layers.

In Fig. 7.4, we also compare the downstream prediction performance from object-centric and distributed representations. We observe that VAE representations tend to achieve lower scores in downstream prediction, although not always by a large margin. In particular, color and size in CLEVR and color in Tetrominoes are predicted relatively well, and significantly better than the baseline. On the other hand, in many cases where VAE representations perform well, they have in fact a considerable advantage if we take the baselines into account (scale in Multi-dSprites, color in Shapestacks, x and y in CLEVR, Multi-dSprites, and Tetrominoes). Moreover, performance from distributed representations often does not improve significantly when using a larger downstream model (see Fig. C.6). In conclusion, although the two classes of representations are difficult to compare on this task, these results suggest that the quantities of interest are present in the VAE representations, but they appear to be less explicit and less easily usable.

Finally, we investigate the relationship between downstream performance and evaluation metrics. Fig. 7.5 shows the Spearman rank correlation of the segmentation and reconstruction metrics with the test performance of downstream predictors. For all datasets and object properties, downstream performance is strongly correlated with the ARI. On the other hand, SC and mSC exhibit inconsistent trends across datasets. Models that correctly separate objects according to the ARI are therefore useful for downstream object property prediction, confirming Hypothesis 1. Downstream pre-

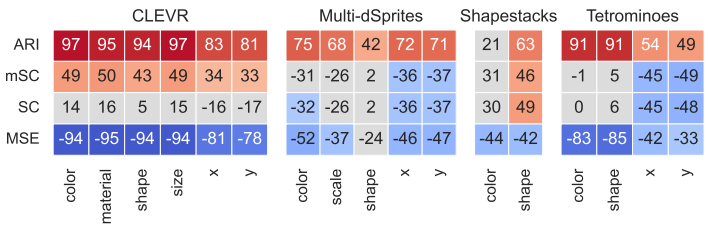


Figure 7.5. Spearman rank correlations between evaluation metrics and downstream performance with an MLP. The correlations are color-coded only when $p<0.05$.

diction performance is also significantly correlated with the reconstruction MSE in all datasets. This is not particularly surprising, since the representation of a model that cannot properly reconstruct the input might not contain the information necessary for property prediction. However, the correlation is generally stronger with the ARI than with the MSE, suggesting that having a notion of objects is more important for downstream tasks than reconstruction accuracy. This is consistent with the findings by Papa, Winther, and Dittadi (2022), where the ARI still correlates strongly with downstream performance when objects have complex textures, while the MSE does not. When segmentation masks are available for validation, ARI should therefore be the preferred metric to select useful representations for downstream tasks. Fig. C.7 in Appendix C.4 shows analogous results for mask matching and for the three other downstream models—these results are broadly similar, except that correlations with ARI tend to be stronger when using mask matching (perhaps unsurprisingly) or larger downstream models.

Summary: Models that accurately segment objects allow for good downstream prediction performance. Despite often having an advantage, distributed representations generally perform worse, but not always significantly: the information is present but less easily accessible. The ARI is consistently correlated with downstream performance, and is therefore useful for model selection when masks are available. The MSE can be a practical unsupervised alternative on these datasets, but it may be less robust on complex textures.

7.4.3 Generalization with one OOD object (Hypothesis 2)

To test Hypothesis 2, we construct settings where a single object is OOD and the others are ID. We change the object style with neural style transfer, change the color of one object at random (only in CLEVR, Tetrominoes, and Shapestacks), or introduce a new shape (only in Multi-dSprites). The unsupervised models are always trained on the original datasets. Then we train downstream models to predict the object properties from the learned representations. We consider two scenarios for this task: (1) train the predictors on the original datasets and test them on the variants with a modified object, (2) train and test the predictors on each variant. In both cases, we test the predictors on representations that might be inaccurate, because the representation function (encoder) is OOD. However, since in case (2) the downstream model is *trained* under distribution shift, this experiment quantifies the extent to

which the representation can still be used by a downstream task that is allowed to adapt to the shift—although the representation might no longer represent objects faithfully, it could still contain useful information.

For Desideratum 1, we observe in Fig. 7.6 that the models are generally robust to distribution shifts affecting a single object. Introducing a new color or a new shape typically does not affect segmentation quality (but note Slot Attention on Tetrominoes), while changing the texture of an object via neural style transfer leads to a moderate drop in ARI in some cases. In Fig. C.8 (Appendix C.4) we observe that SC and mSC show a compatible but less pronounced trend, while the MSE more closely mirrors the ARI. We conclude that the encoder is still partially able to separate objects when one object undergoes a distribution shift at test time.

For Desideratum 2, we observe in Fig. 7.7 (left) that property prediction performance for objects that underwent distribution shifts (color, shape, or texture) is often significantly worse than in the original dataset, whereas the prediction of ID objects is largely unaffected. This is in agreement with Hypothesis 2: changes to one object do not affect the representation of other objects, even when these objects are OOD. Extensive results, including further splits and all downstream models, are shown in Fig. C.10 in Appendix C.4. On the right plot in Fig. 7.7, we observe that retraining the downstream models after the distribution shifts does not lead to significant improvements. This suggests that the shifts introduced here negatively affect not only the downstream model, but also the representation itself. This result also holds with different downstream models and with mask matching (see Figs. C.11 and C.13). While in principle we observe a similar trend for VAEs (see e.g. Figs. C.18 and C.19 in Appendix C.4), their performance is often too close to the respective baseline (Fig. 7.4) for a definitive conclusion to be drawn.

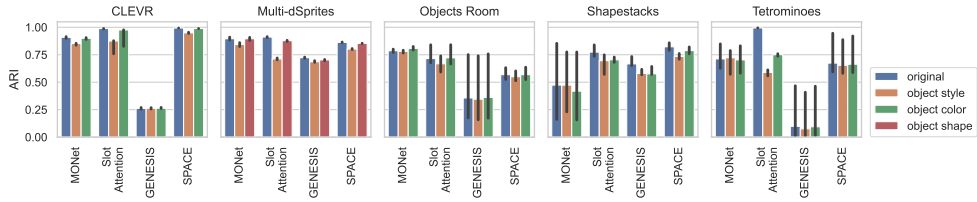


Figure 7.6. Effect of single-object distribution shifts on the ARI. Medians and 95% confidence intervals with 10 random seeds.

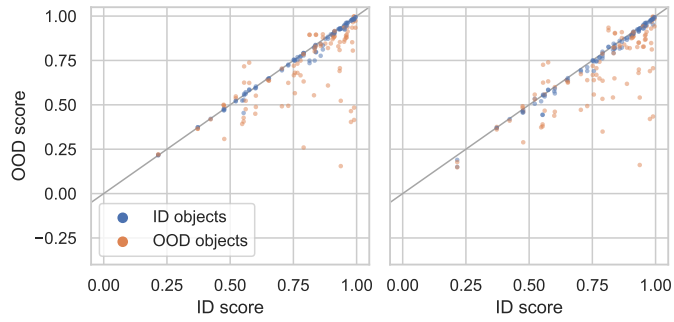


Figure 7.7. ID vs OOD downstream performance with single-object distribution shifts. All datasets, models, and object properties are shown. Metrics: accuracy for categorical attributes, R^2 for numerical attributes. The downstream model (an MLP with one hidden layer) is tested zero-shot out-of-distribution (left) or retrained after the distribution shift has occurred (right).

Summary: The models are generally robust to distribution shifts affecting a single object. Downstream prediction is largely unaffected for ID objects, but may be severely affected for OOD objects. Finally, there seems to be no clear benefit in retraining downstream models after the shifts, indicating that the deteriorated representations cannot easily be adjusted *post hoc*.

7.4.4 Robustness to global shifts (Hypothesis 3)

Finally, we investigate the robustness of object-centric models to transformations changing the global properties of a scene at test time. Here, we consider variants of the datasets with occlusions, cropping, or more objects (only on CLEVR). We train downstream predictors on the original datasets and report their test performance on the dataset variants with global shifts. As before, we also report results of downstream models retrained on the OOD datasets.

For Desideratum 1, Fig. 7.8 shows that segmentation quality is generally only marginally affected by occlusion, but cropping often leads to a significant degradation. In CLEVR, the effect on the ARI of increasing the number of objects is comparable to the effect of occlusions, which suggests that learning about objects is useful for this type of systematic generalization. These trends persist when considering SC and mSC, but appear less pronounced and less consistent across datasets (see Fig. C.9 in Ap-

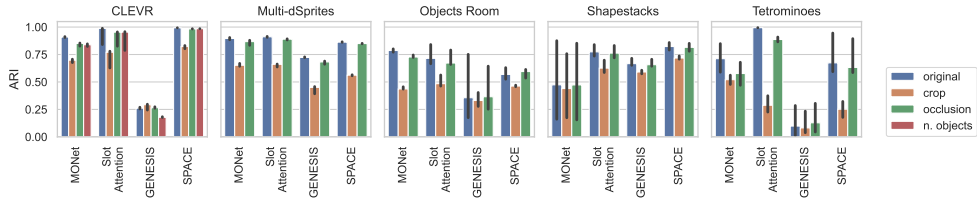


Figure 7.8. Effect of distribution shifts on global scene properties on the ARI. Medians and 95% confidence intervals with 10 seeds.

pendix C.4 for detailed results). As might be expected, when the number of objects is increased in CLEVR, the MSE increases more conspicuously for VAEs than for object-centric models (Fig. C.9, bottom left), likely due to their explicit modeling of objects. However, Fig. C.32 shows that VAEs may, in fact, generalize relatively well to an unseen number of objects, although not nearly as well as some object-centric models.

For Desideratum 2, we train a downstream model on the original dataset and test it under global distribution shifts. These shifts generally have a negative effect on downstream property prediction (Fig. 7.9, left), although this is comparable to the effect on OOD objects when only one object is OOD. This is in agreement with the observation made in Section 7.4.3 that these shifts negatively affect the representation, which is no longer accurate because the encoder is OOD (cf. the "OOD2" scenario

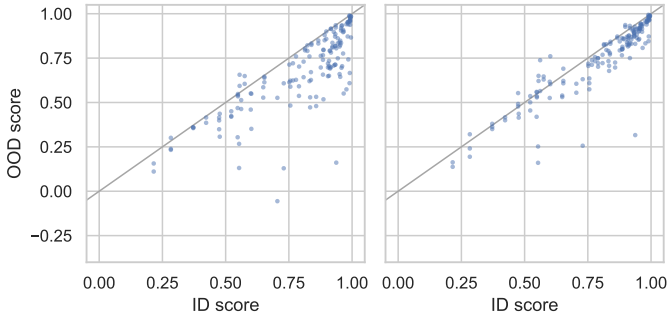


Figure 7.9. ID vs OOD downstream performance with global distribution shifts. All datasets, models, and object properties are shown. Metrics: accuracy for categorical attributes, R^2 for numerical attributes. The downstream model (an MLP with one hidden layer) is tested zero-shot out-of-distribution (left) or retrained after the distribution shift has occurred (right).

in **dittadi2020transfer**). When retraining the downstream models on the OOD datasets while keeping the representation frozen, the performance improves slightly but does not reach the corresponding results on the training distribution (Fig. 7.9, right), as in Section 7.4.3. These observations also hold for different downstream models and with mask matching (Figs. C.14 to C.17), as well as for distributed representations (see, e.g., Fig. C.22) although with similar caveats as in Section 7.4.3.

Summary: The impact of global distribution shifts on the segmentation capability of object-centric models depends on the chosen shift; e.g., cropping consistently has a significant effect. Moreover, the usefulness for downstream tasks decreases substantially in many cases, and the performance of downstream prediction models cannot be satisfactorily recovered by retraining them.

7.5 Other related work

Recent years have seen a number of systematic studies on disentangled representations (Locatello et al., 2019a; b; Steenkiste et al., 2019; Träuble et al., 2021), some of which focusing on their effect on generalization (Dittadi et al., 2021b; 2022b; Esmaeili et al., 2019b; Gondal et al., 2019; Montero et al., 2021). In the context of object-centric learning, Engelcke, Jones, and Posner (2020) investigate their reconstruction bottlenecks to understand how these models can separate objects from the input in an unsupervised manner. In contrast, we specifically test some key implications of learning object-centric representations.

Slot-based object-centric models can be classified according to their approach to separating the objects at a representational level (Greff, Steenkiste, and Schmidhuber, 2020). In models that use *instance slots* (Chen, Artières, and Denoyer, 2019a; Goyal et al., 2019; Greff et al., 2019; 2016; Greff, Steenkiste, and Schmidhuber, 2017; Huang et al., 2020; Kipf et al., 2021; Kipf, Pol, and Welling, 2019; Le Roux et al., 2011; Locatello et al., 2020d; Löwe et al., 2020; Racah and Chandar, 2020; Steenkiste et al., 2018; 2020; Yang, Chen, and Soatto, 2020), each slot is used to represent a different part of the input. This introduces a routing problem, because all slots are identical but they cannot all represent the same object, so a mechanism needs to be introduced to allow slots to communicate with each other. In models based on *sequential slots* (Burgess et al., 2019; Engelcke et al., 2020; Engelcke, Parker Jones, and Posner, 2021; Eslami et al., 2016; Kosiorek et al., 2018; Kossen et al., 2019; Stelzner,

Peharz, and Kersting, 2019), the representational slots are computed in a sequential fashion, which solves the routing problem and allows to dynamically change the number of slots, but introduces dependencies between slots. In models based on *spatial slots* (Crawford and Pineau, 2019; 2020; Deng et al., 2021; Dittadi and Winther, 2019; Jiang* et al., 2020; Lin et al., 2020a; b; Nash et al., 2017), a spatial coordinate is associated with each slot, introducing a dependency between slot and spatial location. In this work, we focus on four scene-mixture models as representative examples of approaches based on instance slots (Slot Attention), sequential slots (MONet and GENESIS), and spatial slots (SPACE).

7.6 Conclusions

In this paper, we identify three key hypotheses in object-centric representation learning: learning about objects is useful for downstream tasks, it facilitates strong generalization, and it improves overall robustness to distribution shifts. To investigate these hypotheses, we re-implement and systematically evaluate four state-of-the-art unsupervised object-centric learners on a suite of five common multi-object datasets. We find that object-centric representations are generally useful for downstream object property prediction, and downstream performance is strongly correlated with segmentation quality and reconstruction error. Regarding generalization, we observe that when a single object undergoes distribution shifts the overall segmentation quality and downstream performance for in-distribution objects is largely unaffected. Finally, we find that object-centric models can still relatively robustly separate objects even under global distribution shifts. However, this may depend on the specific shift, and downstream performance appears to be more severely affected.

An interesting avenue for future work is to continue our systematic investigation of object-centric learning on more complex data with diverse textures, as well as a wide range of more challenging downstream tasks. Furthermore, it would be interesting to compare object-centric and non-object-centric models more fairly: while learning about objects offers clear advantages, the full potential of distributed representations in this context is still not entirely clear, particularly when scaling up datasets and models. Finally, while we limit our study to unsupervised object discovery, it would be relevant to consider methods that leverage some form of supervision when learning about objects. We believe our benchmarking library will facilitate progress along these and related lines of research.

Acknowledgements

We would like to thank Thomas Brox, Dominik Janzing, Sergio Hernan Garrido Mejia, Thomas Kipf, and Frederik Träuble for useful comments and discussions, and the anonymous reviewers for valuable feedback.

CHAPTER 8

Conclusion

8.1 Summary

The central theme of this thesis is learning representations that reflect the data's underlying structure—e.g., by disentangling the ground-truth generative factors of variation or by separately representing objects in a modular fashion—with little or no supervision. In particular, we focused on the potential usefulness of these representations for learning downstream tasks, where models such as classifiers or reinforcement learning agents are trained downstream of pretrained representation functions. Moreover, learning structured representations holds the promise of better systematic generalization, which is a significant issue in modern deep learning. In this dissertation, we investigated to what extent this may help in practice.

In the first two papers, we noted that disentanglement has been shown to be beneficial for a variety of purposes, but thorough quantitative studies have so far largely focused on (1) toy datasets and (2) simple contrived downstream tasks. In Paper I, we took a step towards a more practically relevant scenario: robotic manipulation. We proposed a new dataset and showed that fully disentangled representations can be learned in this more complex setting by using weak supervision and more expressive neural architectures. We then investigated the role of disentanglement for generalization in simple downstream tasks that consist in predicting the ground-truth factors of variation. In Paper II, we extended this work to challenging robotic tasks, and studied the relationship between properties of the pretrained representations, the generalization of simple downstream tasks, and the generalization of downstream reinforcement learning agents.

We found that the the quality of the representations is generally affected by distribution shifts in the data, largely due to a lack of robustness of the learned representation function. We called this scenario "OOD2" and showed that training with input noise is a simple but effective strategy to improve encoder robustness. This is beneficial both for simple downstream prediction tasks and for more complex robotic manipulation, and allows for zero-shot sim-to-real transfer in both cases. Crucially, we did not find evidence of other representation properties being particularly helpful for this generalization scenario.

In contrast, when the representation function is in distribution but the downstream model is out of distribution, we are effectively *testing the OOD generalization of the downstream model itself* because, in this case, the representations can be assumed to be accurate and meaningful. With this test scenario, which we called "OOD1", we want to answer the following question: *are there properties of the pretrained representations that lead downstream models to be generally robust to systematic distribution shifts?* A positive answer comes from Paper I, where we observed that representation functions that *perfectly* separate the true factors of variation tend to lead to robust downstream models for factor prediction (as long as the encoder remains in distribution). On the other hand, when the representations are not fully disentangled, their degree of entanglement does not seem to affect downstream generalization.

In the more challenging robotic tasks in Paper II, disentanglement does not seem to be beneficial for downstream policies. However, we found that, given a pretrained representation function, downstream factor predictors and reinforcement learning agents generalize in similar ways. This can be very convenient in practice: to obtain a downstream model that is robust to a specific class of distribution shifts, we can pre-select promising upstream representation functions using a simpler proxy task where we mirror the desired generalization scenarios of the target downstream task.

Finally, in Paper III, we considered images with multiple objects and studied the implications of learning object-centric representations, with a particular focus on generalization. We observed that object-centric models that successfully separate objects learn useful representations for simple downstream set-prediction tasks. Regarding generalization, we found that object-centric models are particularly robust to distribution shifts that are in some sense related to the compositional structure of an image—e.g., when the number of objects increases, or when a single object is out of distribution—while the picture appears less clear when the shifts affect the data in a less structured manner.

8.2 Discussion

In this dissertation, we have studied representations that explicitly encode some of the structure found in the data. Disentangled representations (at least in their traditional formulation) separate factors of variation; object-centric representations, which are strictly related to the former, focus on separating objects and are more suitable for multi-object settings. A typical argument in favor of explicitly representing structure in this manner is that it should facilitate systematic generalization, thereby narrowing the gap with human-level intelligence. However, based on the findings and discussions in this dissertation, a few remarks are in order.

First, while it may generally be true that encoding structure is beneficial for generalization, the role of disentanglement appears to depend on both the downstream task and the downstream model. A relevant direction for future work includes validating our results on a broader range of representation learning methods, downstream tasks (e.g., abstract reasoning), and downstream models (e.g., different reinforcement learning algorithms).

Second, in the multi-object setting, unstructured models (variational autoencoders in our case) seem to generalize better than expected, although object-centric approaches still have an edge due to their explicit inductive biases. Since quantitative comparisons between these two classes of models are not entirely fair in our setup, future work should attempt to fill this gap in order to clarify more precisely when and how structured models are more useful than unstructured ones.

Finally, the argument that structured representations benefit generalization is not necessarily valid when the representation function itself is out of distribution (the OOD2 setting), since in that case it may not even be encoding information correctly. In fact, we have shown in a variety of settings—both disentanglement and object-centric learning; both in property prediction and in robotic tasks; and under various distribution shifts—that the main bottleneck for OOD2 generalization is the robustness of the encoder, rather than the format of the representations it computes. On the other hand, our study on object-centric learning in Paper III suggests there may be some useful inductive biases in object-centric models that make them relatively robust to some distribution shifts (e.g., one OOD object). An interesting avenue for future work is to systematically investigate these biases. As an orthogonal but related direction, it would be valuable to study how different choices in the architecture, objective, and

optimization affect the learning of useful and modular object representations, in order to discover inductive biases that could enable scaling object-centric learning to real data with minimal supervision.

Appendices

APPENDIX A

Supplementary material for Chapter 5

A.1 Implementation Details

Training. We train the β-VAEs by maximizing the following objective function:

$$\mathcal{L}_{VAE}^{\beta} = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z})) \leq \log p(\boldsymbol{x})$$

with $\beta > 0$ using the Adam optimizer (Kingma and Ba, 2014) with default parameters. We use a batch size of 64 and train for 400k steps. The learning rate is initialized to 1e-4 and halved at 150k and 300k training steps. We clip the global gradient norm to 1.0 before each weight update. Following Locatello et al. (2019b), we use a Gaussian encoder with an isotropic Gaussian prior for the latent variable, and a Bernoulli decoder. Our implementation of weakly supervised learning is based on Ada-GVAE (Locatello et al., 2020b), but uses a symmetrized KL divergence:

$$\tilde{D}_{\mathrm{KL}}(p, q) = \frac{1}{2} D_{\mathrm{KL}}(p\|q) + \frac{1}{2} D_{\mathrm{KL}}(q\|p)$$

to infer which latent dimensions should be aggregated.

The noise added to the encoder's input consists of two independent components, both iid Gaussian with zero mean: one is independent for each subpixel (RGB) and has standard deviation 0.03, the other is a 8×8 pixel-wise (greyscale) noise with standard deviation 0.15, bilinearly upsampled by a factor of 16. The latter has been designed (by visual inspection) to roughly mimic observation noise in the real images due to complex lighting conditions.

Neural architecture. Architectural details are provided in Tables A.1 and A.2, and Fig. A.1 provides a high-level overview. In preliminary experiments, we observed that batch normalization, layer normalization, and dropout did not significantly affect performance in terms of ELBO, model samples, and disentanglement scores, both in the unsupervised and weakly supervised settings. On the other hand, layer normalization before the posterior parameterization (last layer of the encoder) appeared to be beneficial for stability in early training. While using an architecture based on residual blocks leads to fast convergence, in practice we observed that it may be challenging to keep the gradients in check at the beginning of training.[1] In order to solve this issue, we resorted to a simple scalar gating mechanism in the residual blocks (Bachlechner et al., 2020) such that each residual block is initialized to the identity.

Datasets and OOD evaluation. Because we evaluate OOD generalization in terms of cube color hue (except in the sim2real case), we first sampled 8 color hues at random from the 12 specified in Table 5.1. The chosen hues are: $[0°, 120°, 150°, 180°, 210°, 270°, 300°, 330°]$. Then, the dataset D used for training VAEs is generated by randomly sampling values for the factors of variation from Table 5.1, with the color hue restricted to the above-mentioned values. This makes OOD2 evaluation possible, specifically OOD2-A where the learned predictors are tested on representations extracted from images with held-out values of the cube hue.

For evaluation of out-of-distribution generalization, we train the downstream predictors on a subset $D_1 \subset D$ of the representation training set. The downstream training set D_1 is sampled at random from D but only contains a (not necessarily proper) subset of the 8 cube colors. This subset contains 1 color in the OOD1-A case, 4 colors in OOD1-B and OOD1-C, all 8 colors in OOD2 (in this case D_1 is simply a random subset of D). Then we test the downstream predictors on a set D_2 distributionally different from D_1 in terms of cube color (all OOD1 scenarios as well as OOD2-A) or sim2real (OOD2-B). In the OOD1 case, D_2 is also a subset of D and is generated the same way. In each OOD1 case, the test set D_2 is paired with its corresponding D_1 that was used to train the downstream predictors. D_2 contains all colors in D minus those in D_1. In the OOD2-A case, D_2 is a separate dataset containing 5k simulated images like those in D, except that these only contain the 4 colors that were left out from the VAE training set D (hue in $[30°, 60°, 90°, 240°]$). In the OOD2-B case, the

[1] This instability may also be exacerbated in probabilistic models by the sampling step in latent space, where a large log variance causes the decoder input to take very large values. Intuitively, this might be a reason why layer normalization before latent space appears to be beneficial for training stability.

set D_2 is the dataset of real images. Following previous work (e.g. the GBT10000 metric in Locatello et al. (2019b)), the training set D_1 and test set D_2 for downstream tasks contain 10k and 5k images, respectively, except in the OOD2-B case, where the size is limited by the size of the real dataset.

Table A.1. Encoder (left) and decoder (right) architectures. The latent space dimensionality is denoted by d, and $K = 3$ indicates the number of image channels. Last line in the encoder architecture: the fully connected layer parameterizing the log variance of the approximate posterior distributions of the latent variables has custom initialization. The weights are initialized with 1/10 standard deviation than the default value, and the biases are initialized to -1 instead of 0. Empirically, this together with (learnable) LayerNorm was beneficial for training stability at the beginning of training.

Encoder	
Operation	**Output Shape**
Input	$128 \times 128 \times K$
Conv 5x5, stride 2, 64 ch.	$64 \times 64 \times 64$
LeakyReLU(0.02)	—
2x ResidualBlock(64)	—
Conv 1x1, 128 channels	$64 \times 64 \times 128$
AveragePool(2)	$32 \times 32 \times 128$
2x ResidualBlock(128)	—
AveragePool(2)	$16 \times 16 \times 128$
2x ResidualBlock(128)	—
Conv 1x1, 256 channels	$16 \times 16 \times 256$
AveragePool(2)	$8 \times 8 \times 256$
2x ResidualBlock(256)	—
AveragePool(2)	$4 \times 4 \times 256$
2x ResidualBlock(256)	—
Flatten	4096
LeakyReLU(0.02)	—
FC(512)	512
LeakyReLU(0.02)	—
LayerNorm	—
2x FC(d)	$2d$

Decoder	
Operation	**Output Shape**
Input	d
FC(512)	512
LeakyReLU(0.02)	—
FC(4096)	4096
Reshape	$4 \times 4 \times 256$
2x ResidualBlock(256)	—
BilinearInterpolation(2)	$8 \times 8 \times 256$
2x ResidualBlock(256)	—
Conv 1x1, 128 channels	$8 \times 8 \times 128$
BilinearInterpolation(2)	$16 \times 16 \times 128$
2x ResidualBlock(128)	—
BilinearInterpolation(2)	$32 \times 32 \times 128$
2x ResidualBlock(128)	—
Conv 1x1, 64 channels	$32 \times 32 \times 64$
BilinearInterpolation(2)	$64 \times 64 \times 64$
2x ResidualBlock(64)	—
BilinearInterpolation(2)	$128 \times 128 \times 64$
LeakyReLU(0.02)	—
Conv 5x5, K channels	$128 \times 128 \times K$

Table A.2. Architecture of a residual block. The scalar gate is implemented by multiplying the tensor by a learnable scalar parameter before adding it to the block input. Initializing the residual block to the identity by setting this parameter to zero has been originally proposed by Bachlechner et al. (2020). The tensor shape is constant throughout the residual block.

Residual Block
Input: shape $H \times W \times C$
LeakyReLU(0.02)
Conv 3x3, C channels
LeakyReLU(0.02)
Conv 3x3, C channels
Scalar gate
Sum with input

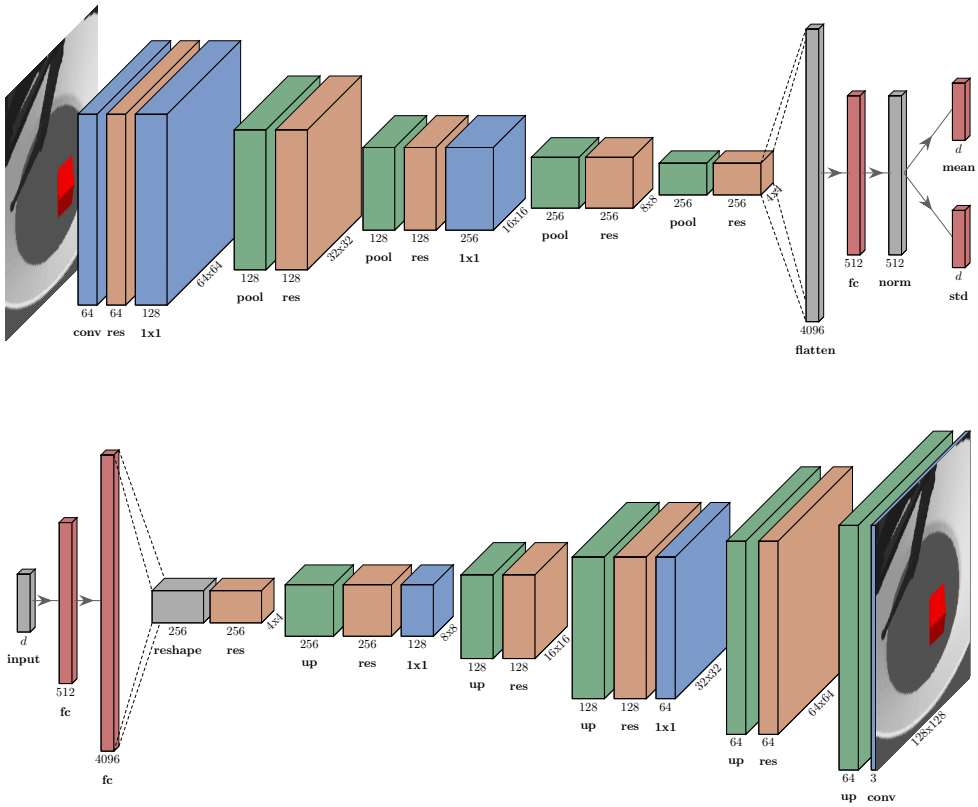


Figure A.1. Schemes of the encoder (top) and decoder (bottom) architectures. In both schemes, information flows left to right. Blue blocks represent convolutional layers: those labeled "conv" have 5x5 kernels and stride 2, while those labeled "1x1" have 1x1 kernels. Each orange block represents a pair of residual blocks (implementation details of a residual block are provided in Table A.2). Green blocks in the encoder represent average pooling with stride 2, and those in the decoder denote bilinear upsampling by a factor of 2. Red blocks represent fully-connected layers. The block labeled "norm" indicates layer normalization. Dashed lines denote tensor reshaping.

A.2 Additional Results

A.2.1 Dataset Correlations

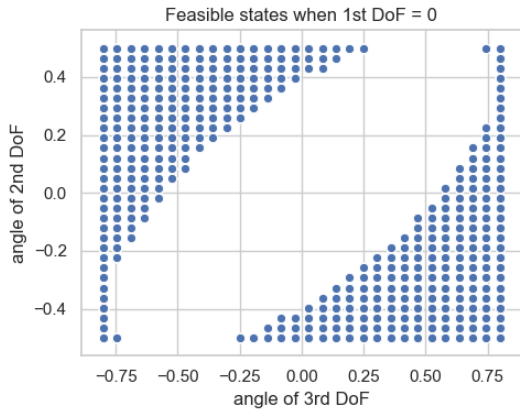


Figure A.2. Feasible states of the 2nd and 3rd DoF when the angle of the 1st DoF is 0. Angles are in radians.

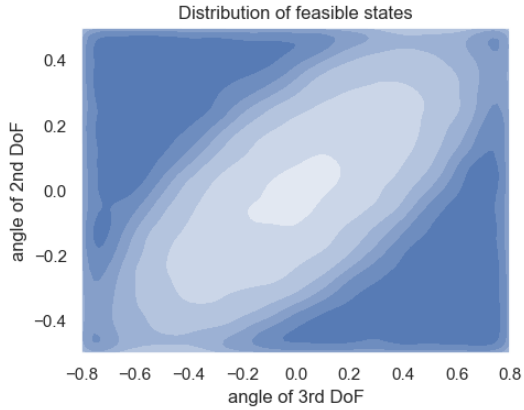


Figure A.3. Density of feasible states of 2nd and 3rd DoF over the whole training dataset. Darker shades of blue indicate regions of higher density. Angles are in radians.

A.2.2 Samples and Reconstructions

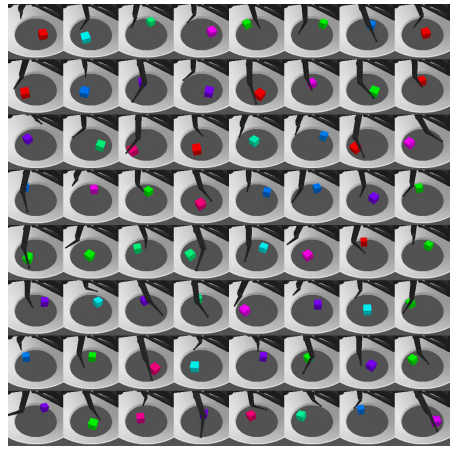


Figure A.4. Samples generated by a trained model (selected based on the ELBO).

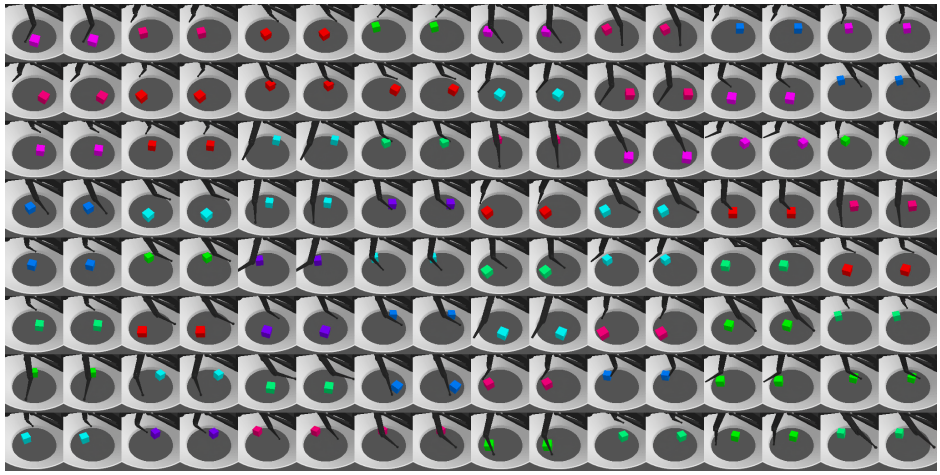


Figure A.5. Input reconstructions by a trained model. This model was selected based on the ELBO. Image inputs are on odd columns, reconstructions on even columns.

A.2.3 Latent Traversals

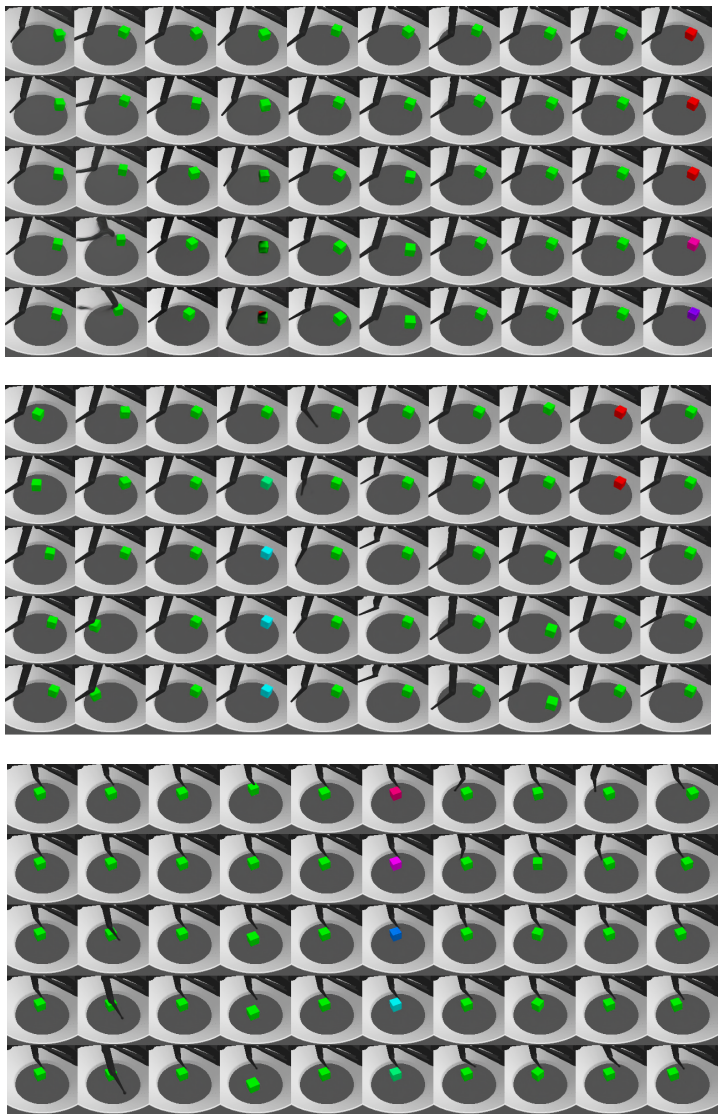


Figure A.6. From top to bottom: latent traversals for a model with low (0.15), medium (0.5), and high (1.0) DCI score.

A.2.4 Unsupervised Metrics and Disentanglement

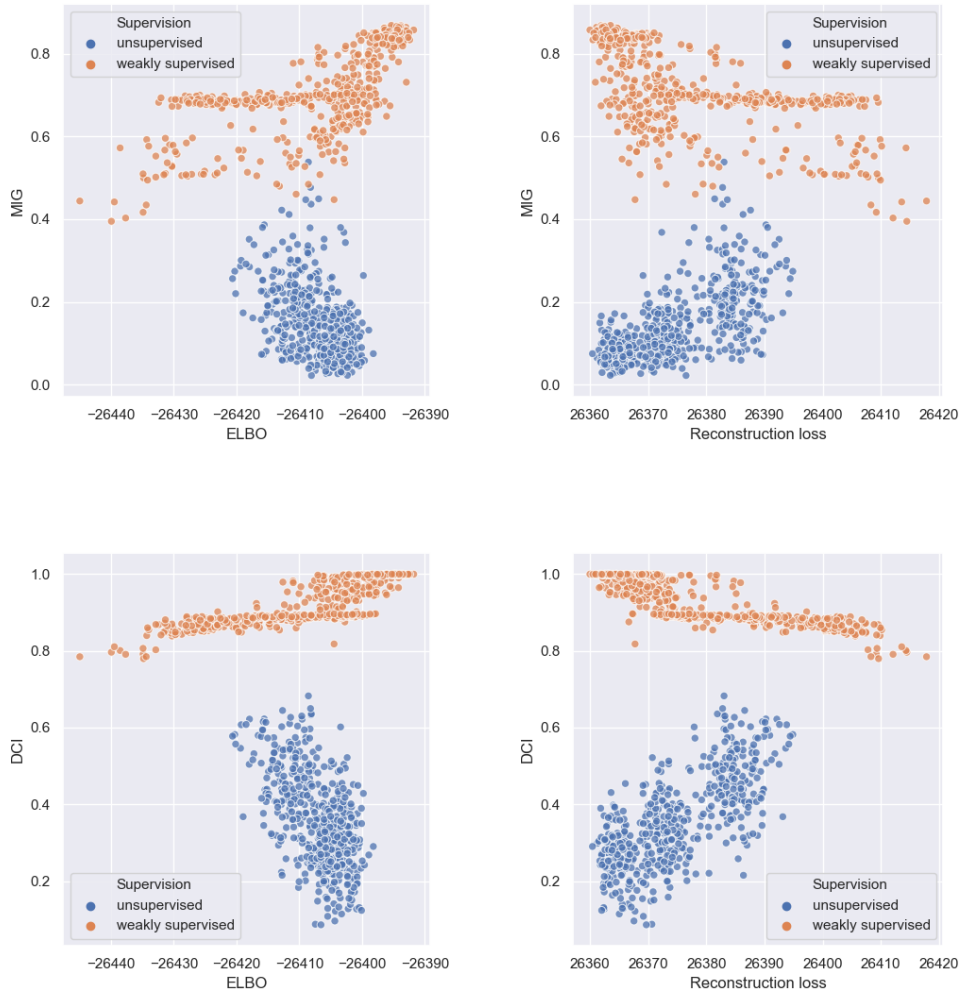


Figure A.7. Scatter plots of unsupervised metrics (left: ELBO; right: reconstruction loss) vs disentanglement (top: MIG; bottom: DCI) for 1,080 trained models, color-coded according to supervision. Each point represents a trained model.

A.2.5 Out-of-Distribution Transfer

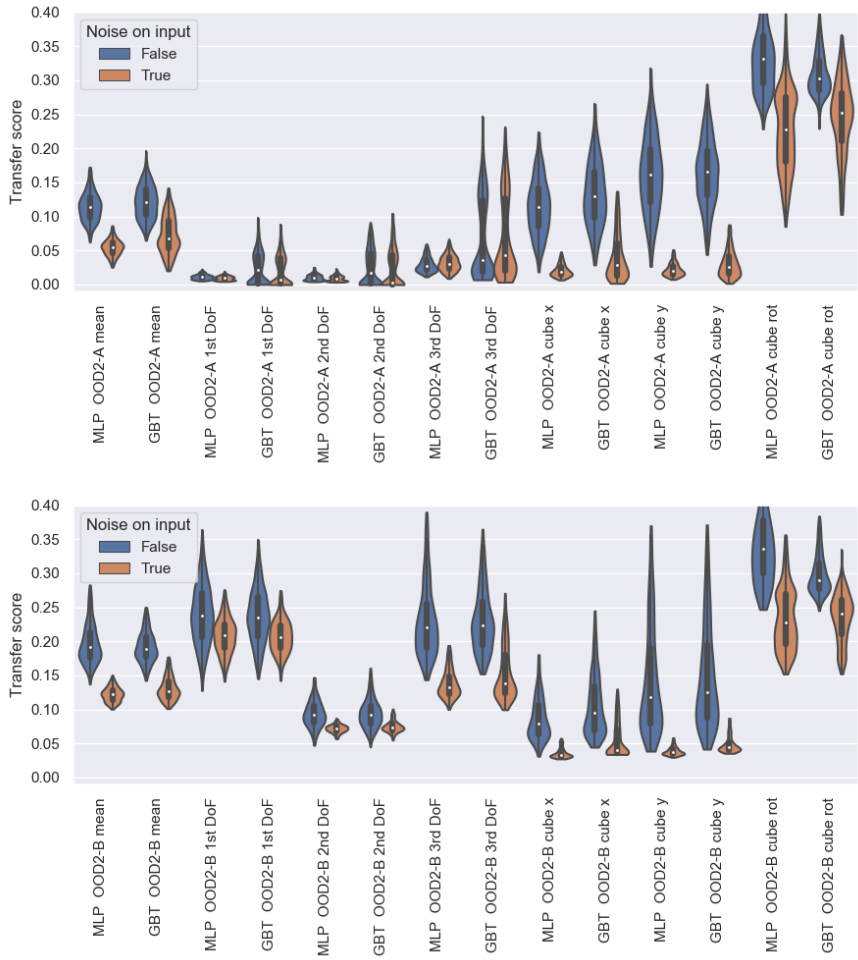


Figure A.8. Transfer metric in OOD2-A (top) and OOD2-B (bottom) settings, decomposed according to the factor of variation and presence of input noise. When noise is added to the input during training, the inferred cube position error is relatively low (the scores are the mean absolute error, and they are normalized to $[0, 1]$). This is particularly useful in the OOD2-B setting (real world) where the joint state is anyway considered known, while object position has to be inferred with tracking methods.

A.2.6 Out-of-Distribution Reconstructions

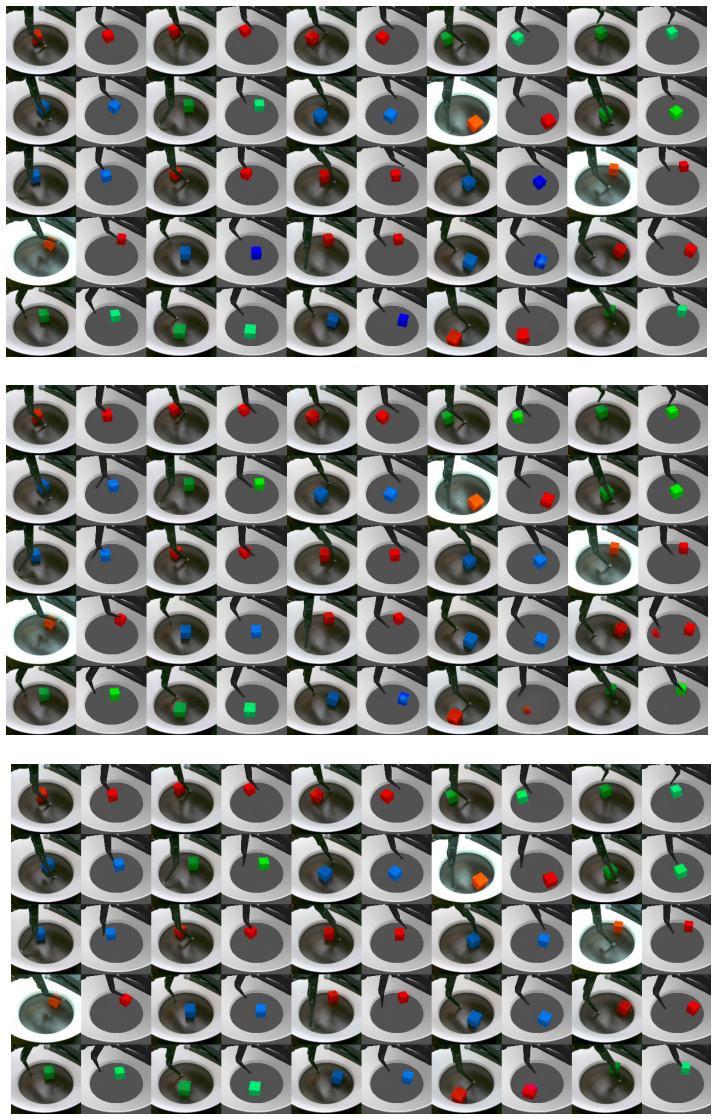


Figure A.9. From top to bottom: reconstructions of real-world images (OOD2-B) for a model with low (0.15), medium (0.5), and high (1.0) DCI score.

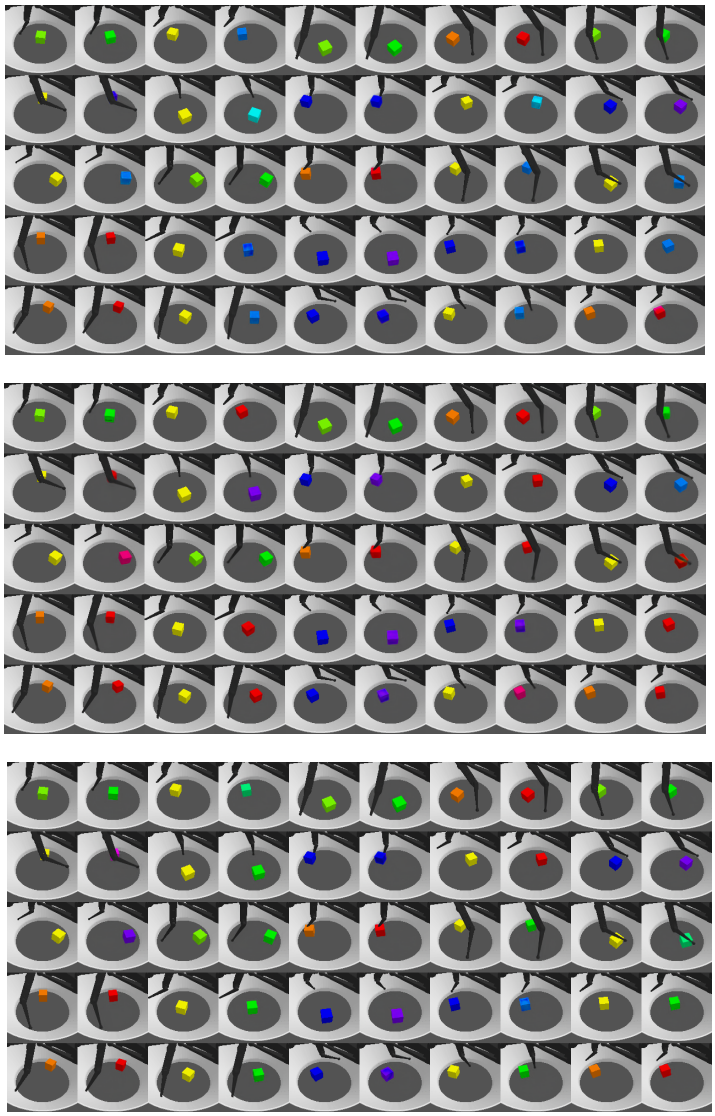


Figure A.10. Reconstructions of simulated images with out-of-distribution encoder (OOD2-A) for a model with low (0.15), medium (0.5), and high (1.0) DCI score.

APPENDIX B

Supplementary material for Chapter 6

B.1 Implementation details

Task definitions and reward structure. We derive both tasks, *object reaching* and *pushing*, from the CausalWorld environments introduced by Ahmed et al. (2021). We pretrain representations on the dataset introduced by Dittadi et al. (2021b), and allow only one finger to move in our RL experiments. We introduce the *object reaching* environment that involves an unmovable cube. We used reward structures similar to those in Ahmed et al. (2021):

- *object reaching*: $r_t = -750\,[d(g_t, e_t) - d(g_{t-1}, e_{t-1})]$
- *pushing*: $r_t = -750\,[d(o_t, e_t) - d(o_{t-1}, e_{t-1})] - 250\,[d(o_t, g_t) - d(o_{t-1}, g_{t-1})] + \rho_t$

where t denotes the time step, $\rho_t \in [0, 1]$ is the fractional overlap with the goal cube at time t, $e_t \in \mathbf{R}^3$ is the end-effector position, $o_t \in \mathbf{R}^3$ the cube position, $g_t \in \mathbf{R}^3$ the goal position, and $d(\cdot, \cdot)$ denotes the Euclidean distance. The cube in *object reaching* is fixed, i.e. $o_t = g_t$ for all t. The time limit is 2 seconds in *object reaching* and 4 seconds in *pushing*.

Success metrics. Besides the accumulated reward along episodes, that is determined by the reward function, we also report two reward-independent normalized success definitions for better interpretability: In *object reaching*, the success metric

indicates progress from the initial end effector position to the optimal distance from the center of the cube. It is 0 if the final distance is greater than or equal to the initial distance, and 1 if the end effector is touching the center of a face of the cube. In *pushing*, the success metric is defined as the volumetric overlap of the cube with the goal cube, and the task can be visually considered solved with a score around 80%. We observed that accumulated reward and success are very strongly correlated, thus allowing us to use one or the other for measuring performance.

Training and evaluation details. During training, the goal position is randomly sampled at every episode. Similarly, the object color is sampled from the 4 specified training colors from $\mathcal{D}_1$ that correspond to the OOD1-B split from Dittadi et al. (2021b).

For each policy evaluation (in-distribution and out-of-distribution variants), we average the accumulated reward and final success over 200 episodes with randomly sampled cube positions and the respective object color in both tasks.

SAC implementation. Our implementation of SAC builds on `stable-baselines`, a Python package introduced by Hill et al. (2018). We use the same SAC hyperparameters used for pushing in Ahmed et al. (2021). When using L1 regularization, we add to the loss function the L1 norm of the first layers of all MLPs, scaled by a coefficient α. We gradually increase regularization by linearly annealing α from 0 to $5 \cdot 10^{-7}$ over 200,000 time steps in *object reaching*, and from 0 to $6 \cdot 10^{-8}$ over 3,000,000 time steps in *pushing*.

B.2 Additional results

B.2.1 Training environment

Fig. 6.2 in Section 6.4.1 shows correlations of unsupervised and supervised metrics with in-distribution reward for *object reaching* and *pushing*, only in the case without regularization. In Fig. B.1 we also show these results in the case with regularization, as well as when adjusting for MLP informativeness.

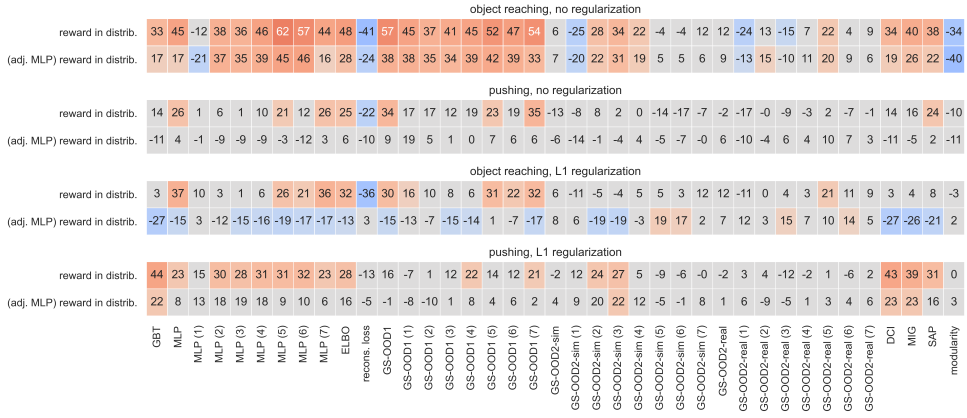


Figure B.1. Rank correlations between metrics and in-distribution reward, with and without adjusting for informativeness. Correlations are color-coded as described in Fig. 6.2.

B.2.2 Out-of-distribution generalization in simulation

In Section 6.4.2 we discussed rank-correlations of OOD rewards with unsupervised, informativeness and generalization scores on *object reaching* without regularization. In Fig. B.2 we also show these results for the case with regularization and on *pushing*, as well as when adjusting for MLP informativeness. Without regularization, we observe on *pushing* very similar correlations along all metrics as we observed on *object reaching*, confirming our conclusions on this more complex task. When using regularization, rank correlations are very similar across both tasks. Interestingly, the correlation between GS-OOD2 scores and OOD2 generalization of the policy is even stronger when using L1 regularization. In contrast to our observations without regularization, we find that the correlation between GS-OOD1 and OOD1 generalization of the policy vanishes when adjusting for the MLP metric.

B.2.2.1 Disentangled representations

As discussed in Section 6.4.2 for *object reaching* without regularization, we observe in Fig. B.2 a weak correlation between some disentanglement metrics and OOD1 reward, which however vanishes when adjusting for MLP informativeness. In agreement with Dittadi et al. (2021b), we observe no significant correlation between disentanglement

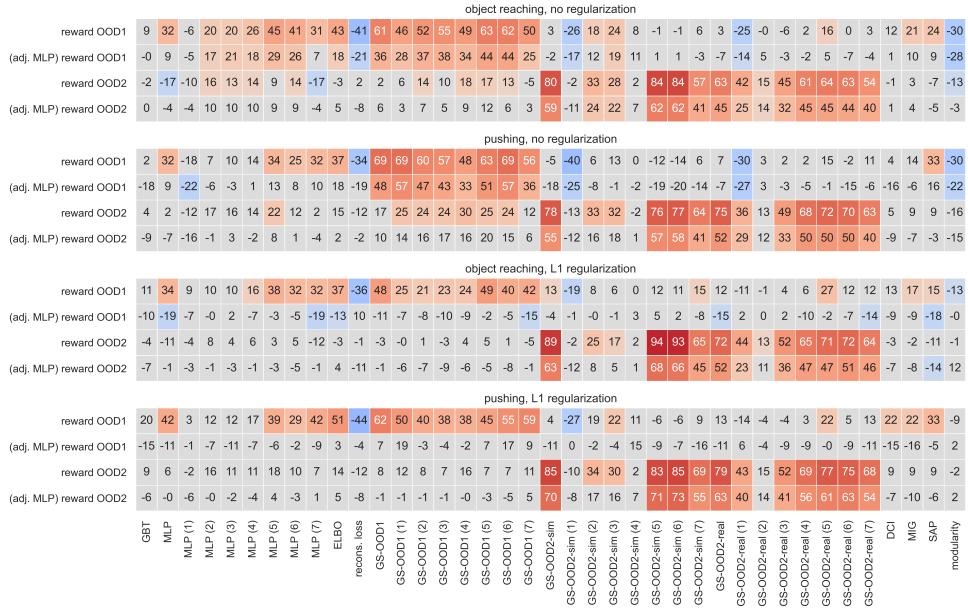


Figure B.2. Rank correlations between metrics and OOD reward, with and without adjusting for informativeness. Correlations are color-coded as described in Fig. 6.2.

and OOD2 generalization, for both tasks, with and without regularization. From Fig. B.3 we see that in some cases, especially without regularization, a very high DCI score seems to lead to better performance on average. However, this behavior is not significant (within error bars), as opposed to the results shown in simpler downstream tasks by Dittadi et al. (2021b). Furthermore, this trend is likely due to representation informativeness, since the correlations with disentanglement disappear when adjusting for the MLP score, as discussed above.

B.2.2.2 Regularization

As seen in Fig. B.3, regularization generally has a positive effect on OOD1 and OOD2 generalization, which is particularly prominent in the OOD1 setting. On the other hand, it leads to lower training rewards both in *object reaching* and in *pushing*. In the latter, the performance drop is particularly significant, while in *object reaching* it is negligible.

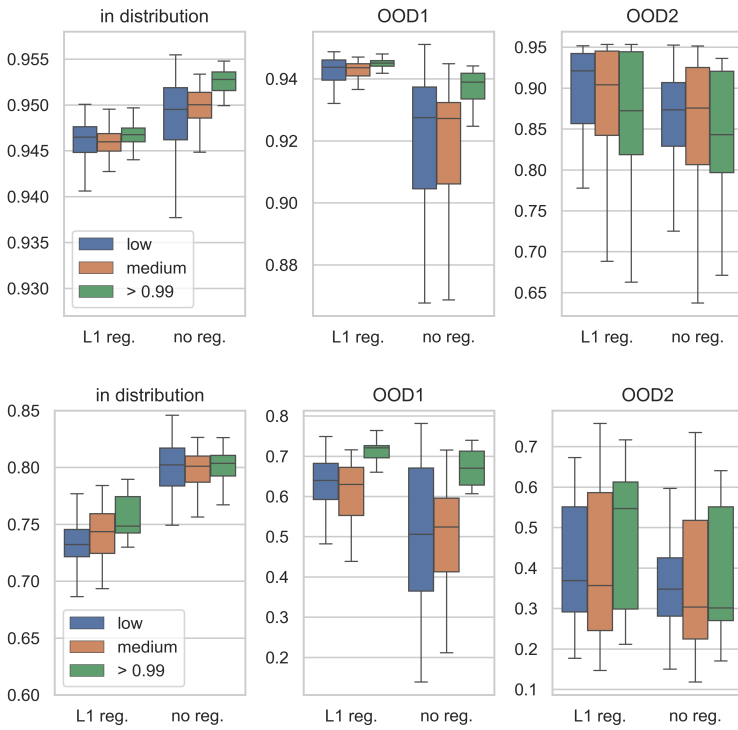


Figure B.3. Fractional success on *object reaching* (**top**) and *pushing* (**bottom**), split according to low (blue), medium-high (orange), and almost perfect (green) disentanglement. Results for *object reaching* are also reported in Fig. 6.5 in Section 6.4.2.

B.2.2.3 Sample efficiency

In addition to the analysis reported in the main paper, we investigate how representation properties affect sample efficiency. Specifically, we store checkpoints of our policies at $t \in \{20\text{k}, 50\text{k}, 100\text{k}, 400\text{k}\}$ for *object reaching* and $t \in \{200\text{k}, 500\text{k}, 1\text{M}, 3\text{M}\}$ for *pushing*. We then evaluate policies at these time step on the same three environments as before: (1) on the cube colors from training; (2) on the OOD1 cube colors; and (3) on the OOD2-sim cube colors. Results are summarized in Fig. B.4 for *object reaching* and Fig. B.5 for *pushing*.

On *object reaching* (Fig. B.4), we observe very similar trends with and without regularization: Unsupervised metrics (ELBO and reconstruction loss) display a correlation

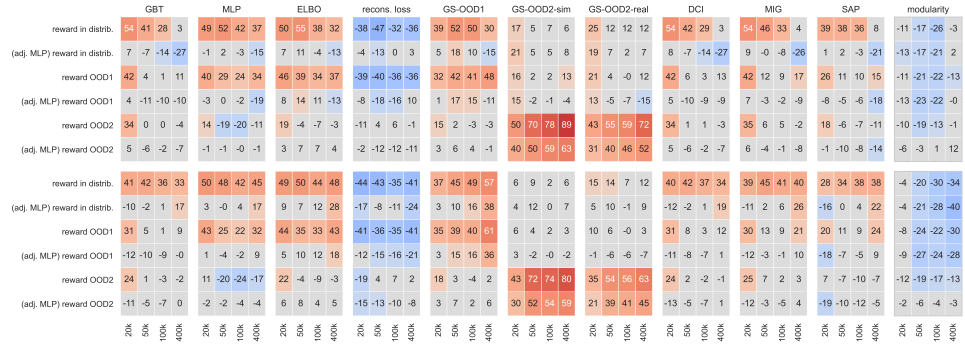


Figure B.4. Sample efficiency analysis for *object reaching*. Rank correlations of rewards with relevant metrics along multiple time steps. Correlations are color-coded as described in Fig. 6.2.

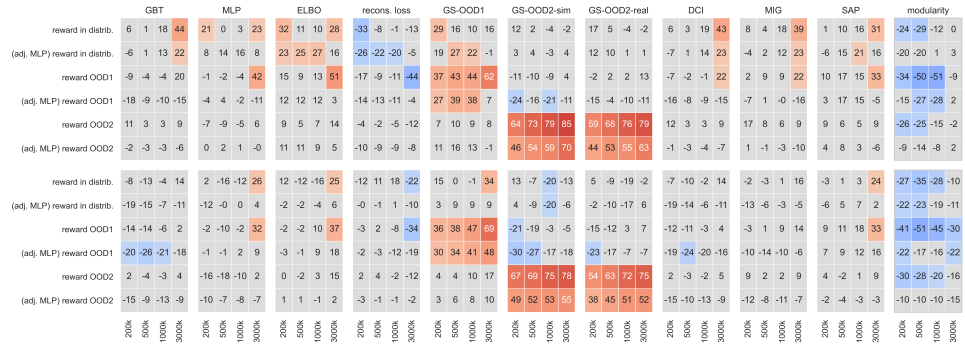


Figure B.5. Sample efficiency analysis for *pushing*. Rank correlations of rewards with relevant metrics along multiple time steps. Correlations are color-coded as described in Fig. 6.2.

with the training reward, as do the supervised informativeness metrics (GBT and MLP). This is strongest on early timesteps, meaning these scores could be important for sample efficiency. Similarly, we observe a correlation with the disentanglement scores DCI, MIG and SAP. With the help of the additional evaluation of rewards adjusted for MLP informativeness, we can attribute this correlation again to this common confounder. Crucially, we see that the generalization scores (GS) are correlated with generalization of the corresponding policies under OOD1 and OOD2 shifts for all recorded time steps, confirming the results in the main text.

On *pushing* (Fig. B.5), many correlations at early checkpoints are significantly reduced, especially with regularization. This behavior might be due to the more complicated nature of the task, which involves learning to reach the cube first, and then push it to the goal. Correlations are primarily seen towards the end of training, with similar spurious correlations with disentanglement as elaborated above. Importantly, correlations between generalization scores (GS) and policy generalization under the same distribution shifts remain strong and statistically significant, corroborating the analysis in the main text.

B.2.2.4 Generalization to a novel shape

As mentioned in Section 6.4.2, on the *object reaching* task, we also test generalization w.r.t. a novel object shape by replacing the cube with an unmovable sphere. This corresponds to a strong OOD2-type shift, since shape was never varied when training the representations. We then evaluate a subset of 960 trained policies as before, with the same color splits. Surprisingly, the policies appear to handle the novel shape as we see from the histograms in Fig. B.6 in terms of success and final distance. In fact, when the sphere has the same colors that the cube had during policy training, *all* policies get closer than 5 cm to the sphere on average, with a mean success metric of about 95%. On sphere colors from the OOD1 split, more than 98.5% move the finger closer than this threshold, and on the strongest distribution shift (OOD2-sim colors and cube replaced by sphere) almost 70% surpass that threshold with an average success metric above 80%.

B.2.3 Deploying policies to the real world

In Fig. B.7 we show three representative episodes of testing a reaching policy on the real robot for the strong OOD shift with a novel sphere object shape instead of the cube from training. We present the respective videos in the project page. There we also present videos of additional real-world episodes on pushing and reaching cubes of different colors.

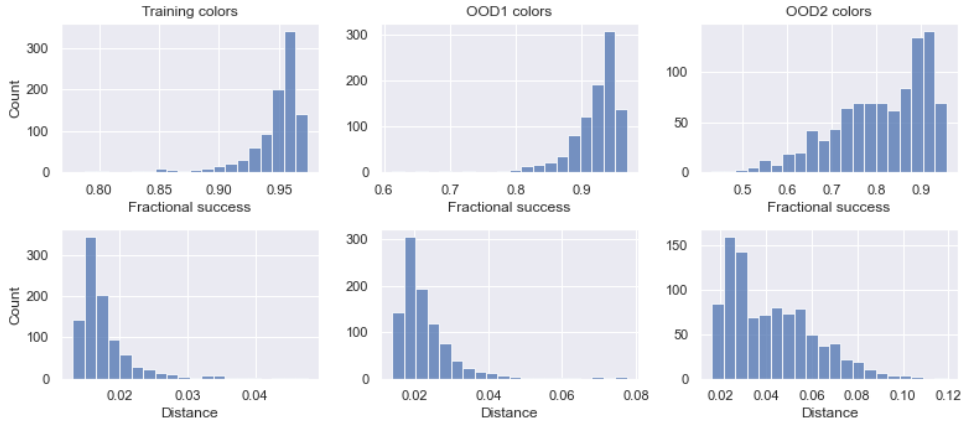


Figure B.6. Testing policies for *object reaching* under the same in-distribution, OOD1, and OOD2 evaluation protocols regarding object color in simulation, but replacing the cube with a sphere, which was never used in training.

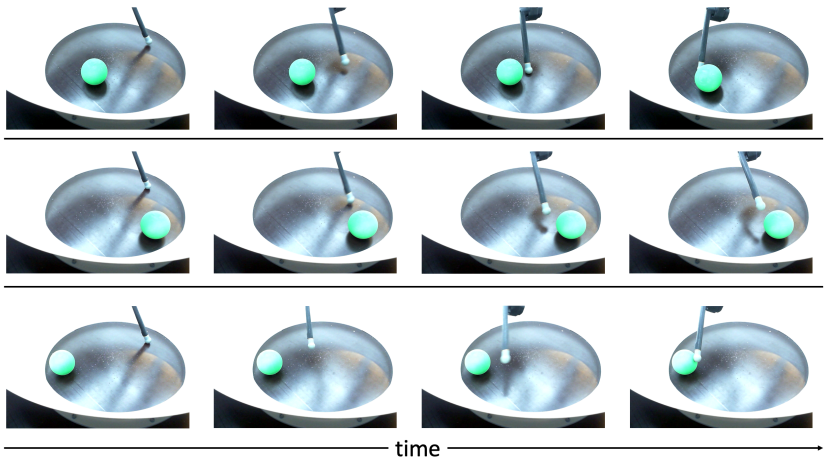


Figure B.7. Transferring policies for *object reaching* to the real robot setup without any fine-tuning on a green sphere (unseen shape *and* color). Correlations are color-coded as described in Fig. 6.2.

APPENDIX C

Supplementary material for Chapter 7

C.1 Models

In this section, we give an informal overview of the models included in this study and provide details on the implementation and hyperparameter choices.

C.1.1 Overview of the models

MONet. In MONet (Burgess et al., 2019), attention masks are computed by a recurrent segmentation network that takes as input the image and the current *scope*, which is the still unexplained portion of the image. For each slot, a variational autoencoder (the *component VAE*) encodes the full image and the current attention mask, and then decodes the latent representation to an image reconstruction and mask. The reconstructed images are combined using the *attention* masks (*not* the masks decoded by the component VAE) into the final reconstructed image. The reconstruction loss is the negative log-likelihood of a spatial Gaussian mixture model (GMM) with one component per slot, where each pixel is modeled independently. The overall training loss is a (weighted) sum of the reconstruction loss, the KL divergence of the component VAEs, and an additional mask reconstruction loss for the component VAEs.

GENESIS. Similarly to MONet, GENESIS (Engelcke et al., 2020) models each image as a spatial GMM. The spatial dependencies between components are modeled by an autoregressive prior distribution over the latent variables that encode the mixing probabilities. From the image, an encoder and a recurrent network are used to compute the latent variables that are then decoded into the mixing probabilities. The mixing probabilities are pixel-wise and can be seen as attention masks for the image. Each of these is concatenated with the original image and used as input to the component VAE, which finds latent representations and reconstructs each scene component. These are combined using the mixing probabilities to obtain the reconstruction of the image. While in MONet the attention masks are computed by a deterministic segmentation network, GENESIS defines an autoregressive prior on latent codes that are decoded into attention masks. GENESIS is therefore a proper probabilistic generative model, and it is trained by maximizing a modification of the ELBO introduced by Rezende and Viola (2018), which adaptively trades off the likelihood and KL terms in the ELBO.

Slot Attention. As our focus is on the object discovery task, we use the autoencoder model proposed in the Slot Attention paper (Locatello et al., 2020d). The encoder consists of a CNN followed by the Slot Attention module, which maps the feature map to a set of slots through an iterative refinement process. At each iteration, dot-product attention is computed with the input vectors as keys and the current slot vectors as queries. The attention weights are then normalized over the slots, introducing competition between the slots to explain the input. Each slot is then updated using a GRU that takes as inputs the current slot vectors and the normalized attention vectors. After the refinement steps, the slot vectors are decoded into the appearance and mask of each object, which are then combined to reconstruct the entire image. The model is optimized by minimizing the MSE reconstruction loss. While MONet and GENESIS use sequential slots to represent objects, Slot Attention employs instance slots.

SPACE. Spatially Parallel Attention and Component Extraction (SPACE) (Lin et al., 2020b) combines the approaches of scene-mixture models and spatial attention models. The foreground objects are segregated using bounding boxes computed through a parallel spatial attention process. The parallelism allows for a larger number of bounding boxes to be processed compared to previous related approaches. The background elements are instead modeled by a mixture of components. The use of bounding boxes for the foreground objects could lead to under- or over-segmentation

if the size of the bounding box is not tuned appropriately. An additional boundary loss tries to address the over-segmentation issue by penalizing splitting objects across bounding boxes.

VAE baselines. We train variational autoencoders (VAEs) (Kingma and Welling, 2014; Rezende, Mohamed, and Wierstra, 2014) as baselines that learn distributed representations. Following Greff et al. (2019), we use two different decoder architectures: one consisting of an MLP followed by transposed convolutions, and one where the MLP is replaced by a broadcast decoder (Watters et al., 2019). The VAEs are trained by maximizing the usual variational lower bound (ELBO).

C.1.2 Implementation details

We implement our library in PyTorch (Paszke et al., 2019). All models are either re-implemented or adapted from available code, and quantitative results from the literature are reproduced, when available. As shown in Table C.1, all methods included in our study were originally evaluated only on a subset of the datasets considered in our study. Thus, the recommended hyperparameters for a given model are likely to be suboptimal in the datasets on which such model was not evaluated. When a model performed particularly bad on a dataset, we attempted to find better hyperparameter values for the sake of the soundness of our study. We provide implementation and training details for each model below.

MONet. We re-implement MONet following the implementation details in Burgess et al. (2019). In order to make this model work satisfactorily on Shapestacks and

Table C.1. Datasets used for quantitative and/or qualitative evaluation in the publications corresponding to the four object-centric models considered in this study. Here we train and evaluate all models on all datasets.

	CLEVR	Multi-dSprites	Objects Room	Shapestacks	Tetrominoes
MONet	✓	✓*	✓		
Slot Attention	✓	✓			✓
GENESIS		✓*	✓	✓	
SPACE					

*These publications use a variant of Multi-dSprites with colored background as opposed to grayscale.

Tetrominoes—the two datasets where MONet was not originally tested—we ran a grid search over hyperparameters on both datasets, as follows:

- Optimizer: Adam or RMSprop, both with default PyTorch parameters.
- $\beta \in \{0.1, 0.5\}$.
- Learning rate in {3e-5, 1e-4}.
- $(\sigma_{\text{bg}}, \sigma_{\text{fg}}) \in \{(0.06, 0.1), (0.12, 0.18), (0.2, 0.24), (0.25, 0.3), (0.3, 0.36)\}$.

A summary of the final hyperparameter choices is shown in Table C.2.

Slot Attention. We re-implement the Slot Attention autoencoder based on the official TensorFlow implementation and the corresponding publication (Locatello et al., 2020d). We mostly use the recommended hyperparameter values and learning rate schedule. On Objects Room and Shapestacks, we use the same parameters as for Multi-dSprites, which has the same resolution. On CLEVR, we make a few changes to accommodate the larger image size. For the decoder, we follow the approach in Locatello et al. (2020d) and use the broadcast decoder from a broadcasted shape of 8×8 rather than 128×128, and use four times a stride of 2 in the decoder. For the encoder, we follow the set prediction architecture in Locatello et al. (2020d) and use two strides of 2 in the encoder. Finally, we use a batch size of 32 rather than 64.

Table C.2. Overview of the main hyperparameter values for MONet. When dataset-specific values are not given, the defaults are used.

Hyperparameter	**Default value**	**Dataset-specific values**		
		CLEVR	**Shapestacks**	**Tetrominoes**
Optimizer	Adam	RMSprop	RMSprop	—
Learning rate	1e-4	3e-5	—	—
Batch size	64	32	—	—
Training steps	500k	—	—	—
σ_{bg}	0.06	—	0.2	0.3
σ_{fg}	0.1	—	0.24	0.36
β	0.5	—	0.1	—
γ	0.5	—	—	—
Latent space size	16	—	—	—
U-Net blocks	5	6	—	4

GENESIS. We re-implement GENESIS based on the official implementation and the corresponding publication (Engelcke et al., 2020), and use the recommended hyperparameter values. On Objects Room, we use the same hyperparameters as described in the paper for Multi-dSprites and Shapestacks, which have the same resolution. On CLEVR, which has 128×128 images, we use an additional stride of 2 in the convolutional layer at the middle of both encoder and decoder (the output padding in the decoder is adjusted accordingly). In this case we also reduce the batch size from 64 to 32. On Tetrominoes (32×32 images), we change the first stride in the encoder and the last stride in the decoder from 2 to 1.

SPACE. We adapt the official PyTorch implementation of SPACE to integrate it in our library. While in Lin et al. (2020b) the authors train SPACE for 160k steps, here we train it for 200k. Since SPACE was not tested on any of the five datasets considered here (see Table C.1), we perform a hyperparameter sweep for all datasets. For each dataset, we run a random search over hyperparameters by training 100 models for 100k steps. Table C.3 shows the random search definition, the hyperparameter values used for each dataset, and how they differ from those used in the original publication for the *3D-Rooms* dataset (although we omit some hyperparameters that we leave unchanged).

VAEs. The architecture details for the VAEs are presented in Tables C.4 to C.6. These are used for Shapestacks, Multi-dSprites, and Objects Room. For CLEVR, an additional ResidualBlock with 64 channels and a AvgPool2D layer is added at the end of the stack of ResidualBlocks, to downsample the image one more time. This is mirrored in the decoder, where a ResidualBlock with 256 channels and a (bilinear) Interpolation layer is added at the beginning of the stack of ResidualBlocks. The same happens in the broadcast decoder case. For Tetrominoes, the number of layers is the same, but the last AvgPool2D layer is removed from the encoder and the first Interpolation layer is removed from the decoder, to have one less downsampling and upsampling, respectively. The latent space size is chosen to be 64 times the number of slots that would be used when training an object-centric model on the same dataset. Note that the default number of slots varies depending on the dataset, as shown in Table C.7.[1]

[1]Here we consider the default for MONet, Slot Attention, and GENESIS, and we disregard SPACE. Although SPACE has a much larger number of slots, this is not comparable with the other models because of the grid-based spatial attention mechanism.

Table C.3. Hyperparameters for SPACE experiments. Here we show: the hyperparameters recommended by Lin et al. (2020b) for the 3D-Rooms dataset on the official code repository; the hyperparameter space considered for our random search; the chosen default values across datasets; the dataset-specific values for CLEVR and Tetrominoes, which override the defaults. We omit some of the hyperparameters that we left unchanged from Lin et al. (2020b).

Hyperparameter	**Original (3D-Rooms)**	**Sweep values**	**Default value**	**Dataset-specific values**	
				CLEVR	**Tetrominoes**
FG optimizer	RMSprop	RMSprop	RMSprop	—	—
FG learning rate	1e-5	{3e-6, 1e-5, 3e-5, 1e-4}	3e-5	1e-4	1e-4
BG optimizer	Adam	Adam	Adam	—	—
BG learning rate	1e-3	1e-3	1e-3	—	—
Batch size	12	$\{16, 32\}$	32	—	—
σ_{bg}	0.15	$\{0.05, 0.15, 0.35\}$	0.15	0.05	—
σ_{fg}	0.15	$\{0.02, 0.05, 0.15, 0.35\}$	0.15	0.05	—
G (FG grid size)	8	$\{4, 8\}$	8	—	4
K (BG n. of slots)	5	$\{1, 5\}$	5	—	—
Boundary loss off step	100k	{20k, 100k}	20k	—	100k
τ anneal end step	20k	{20k, 50k}	50k	20k	—
Mean of $p(\mathbf{z}_{\text{pres}})$ (start/end values)	(0.1, 0.01)	{(0.1, 0.01), (0.5, 0.05)}	(0.5, 0.05)	(0.1, 0.01)	(0.1, 0.01)
Mean of $p(\mathbf{z}_{\text{scale}})$ (start/end values)	$(-1, -2)$	$\{(-1, -2), (0, -1)\}$	$(0, -1)$	—	—

Table C.4. Structure of the encoder for both the vanilla and broadcast VAE, excluding the final linear layer that parameterizes μ and $\log \sigma^2$ of the approximate posterior.

Encoder		
Type	*Size/Ch.*	*Notes*
Input: $\mathbf{x}$	3	
Conv 5×5		Stride 2, Padding 2
LeakyReLU		
Residual Block	64	2 Conv layers
Residual Block	64	2 Conv layers
Conv 1×1	128	
AvgPool2D		Kernel size 2, Stride 2
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
AvgPool2D		Kernel size 2, Stride 2
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
Conv 1×1	256	
Residual Block	256	2 Conv layers
Residual Block	256	2 Conv layers
Flatten		
LeakyReLU		
Linear	512	
LeakyReLU		
LayerNorm		

Table C.5. Structure of the decoder for the vanilla VAE.

Vanilla Decoder		
Type	*Size/Ch.*	*Notes*
Input: $\mathbf{z}$	64× num. slots	
LeakyReLU		
Linear	512	
LeakyReLU		
Unflatten		
Residual Block	256	2 Conv layers
Residual Block	256	2 Conv layers
Conv 1×1	128	
Interpolation		Scale 2
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
Interpolation		Scale 2
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
Conv 1×1	64	
Interpolation		Scale 2
Residual Block	64	2 Conv layers
Residual Block	64	2 Conv layers
Interpolation		Scale 2
LeakyReLU		
Conv 5×5	Image channels	Stride 1, Padding 2

Table C.6. Structure of the decoder for the broadcast VAE. One less Interpolation is required, because the final image size for this architecture is 64 and the broadcasting is to a feature map of size 8.

Broadcast Decoder		
Type	*Size/Ch.*	*Notes*
Input: **z**	64× num. slots	
Broadcast	64× num. slots +2	Broadcast dim. 8
Residual Block	256	2 Conv layers
Residual Block	256	2 Conv layers
Conv 1×1	128	
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
Interpolation		Scale 2
Residual Block	128	2 Conv layers
Residual Block	128	2 Conv layers
Conv 1×1	64	
Interpolation		Scale 2
Residual Block	64	2 Conv layers
Residual Block	64	2 Conv layers
LeakyReLU		
Conv 5×5	Image channels	Stride 1, Padding 2

C.2 Datasets

We collected 5 existing multi-object datasets and converted them into a common format. *Multi-dSprites*, *Objects Room* and *Tetrominoes* are from DeepMind's Multi-Object Datasets collection, under the Apache 2.0 license (Kabra et al., 2019). *CLEVR* was originally proposed by Johnson et al. (2017), with segmentation masks introduced by Kabra et al. (2019). *Shapestacks* was proposed by Groth et al. (2018) under the GPL 3.0 license. Details on these datasets are provided in the following subsections. See Fig. C.1 for sample images and ground-truth segmentation masks for these datasets. In Table C.7, we report dataset splits, number of foreground and background objects, and number of slots used when training object-centric models.

C.2.1 CLEVR

This dataset consists of 128×128 images of 3D scenes with up to 10 objects, possibly occluding each other. Objects can have different colors (8 in total), materials (rubber or metal), shapes (sphere, cylinder, cube), sizes (small or large), x and y positions, and rotations. Objects can be occluded by others. On average, 6.2 objects are visible. As in previous work (Greff et al., 2019; Locatello et al., 2020d), we learn object-centric representations on the CLEVR6 variant, which contains at most 6 objects. There are 100 000 samples in the full dataset, and 53 483 in the CLEVR6 variant (at most 6 objects). The CLEVR dataset has been cropped and resized according to the procedure detailed originally by Burgess et al. (2019).

Each object is annotated with the following properties:

- `color` (categorical): 8 colors:
  - Red. RGB:`[173, 35, 35]`
  - Cyan. RGB:`[41, 208, 208]`
  - Green. RGB:`[29, 105, 20]`
  - Blue. RGB:`[42, 75, 215]`
  - Brown. RGB:`[129, 74, 25]`
  - Gray. RGB:`[87, 87, 87]`
  - Purple. RGB:`[129, 38, 192]`

  - Yellow. RGB:`[255, 238, 51]`
- `material` (categorical): The material of the object: rubber or metal.
- `shape` (categorical): The shape of the object: sphere, cylinder or cube.
- `size` (categorical): The size of the object: small or large.
- `x` (numerical): The x coordinate in 3D space.
- `y` (numerical): The y coordinate in 3D space.

C.2.2 Multi-dSprites

This dataset is based on the *dSprites* dataset (Matthey et al., 2017). Following previous work (Greff et al., 2019; Locatello et al., 2020d), we use the Multi-dSprites variant with colored sprites on a grayscale background. Each scene has 2–5 objects with random shapes (ellipse, square, heart), sizes (6 discrete values in $[0.5, 1]$), x and y position, orientation, and color (randomly sampled in HSV space). Objects can occlude each other. The intensity of the uniform grayscale background is randomly sampled in each image. Images have size 64×64.

Each object is annotated with the following properties:

- `color` (numerical): 3-dimensional RGB color vector.
- `scale` (numerical): Scaling of the object, 6 uniformly spaced values between 0.5 and 1.
- `shape` (categorical): The shape type of the object (ellipse, heart, square).
- `x` (numerical): Horizontal position between 0 and 1.
- `y` (numerical): Vertical position between 0 and 1.

C.2.3 Objects Room

This dataset was originally introduced by Eslami et al. (2018) and consists of 64×64 images of 3D scenes with up to three objects. Since this dataset includes masks but no labels for the object properties, we can use it only to evaluate segmentation performance.

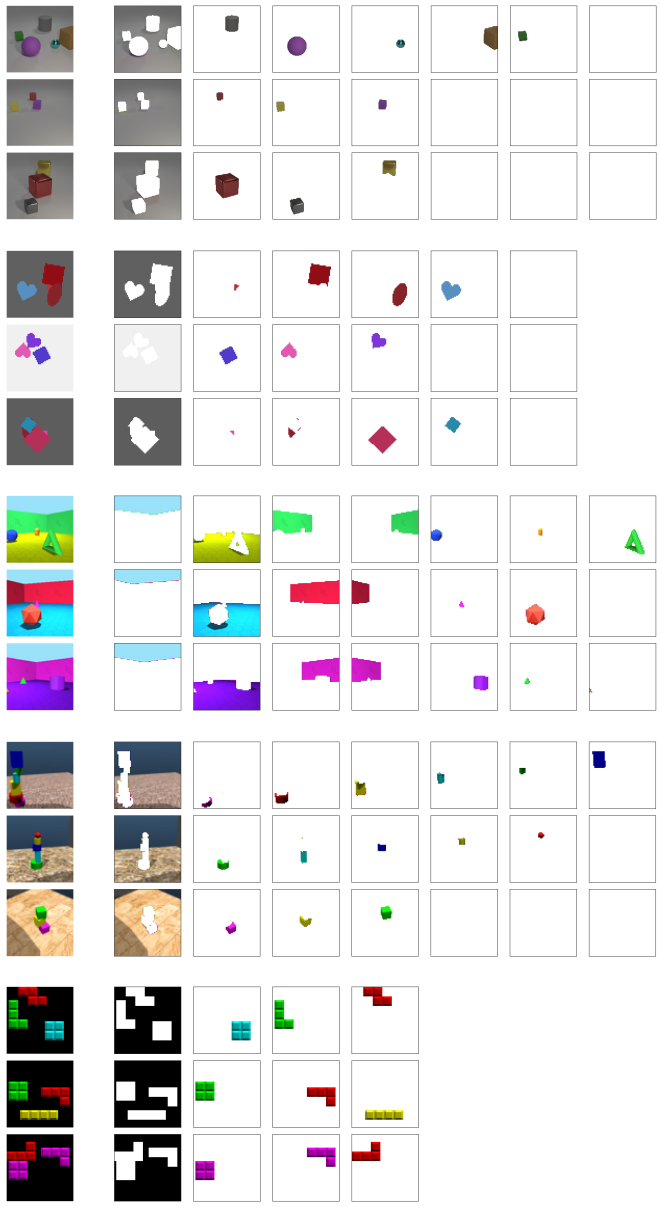


Figure C.1. Examples of images from the datasets considered in this work. The leftmost column represents the original image, the other columns show all the objects in the scene according to the ground-truth segmentation masks. Top to bottom: CLEVR6, Multi-dSprites, Objects Room, Shapestacks, Tetrominoes.

Table C.7. Dataset splits, number of foreground and background objects, and number of slots used when training object-centric models.

Dataset Name	Train Size	Validation Size	Test Size	Background Objects	Foreground Objects	Slots
CLEVR6	49483	2000	2000	1	3–6	7*
Multi-dSprites	90000	5000	5000	1	2–5	6*
Objects Room	90000	5000	5000	4	1–3	7*
Shapestacks	90000	5000	5000	1	2–6	7*
Tetrominoes	90000	5000	5000	1	3	4†

*In SPACE we use 69 slots: 5 background slots, and a grid of 8×8 foreground slots.
†In SPACE we use 21 slots: 5 background slots, and a grid of 4×4 foreground slots.

C.2.4 Shapestacks

This dataset consists of 64×64 images of 3D scenes where objects are stacked to form a tower. Each scene is available under different camera views. Object properties are shape (cube, cylinder, sphere), color (6 possible values), size (numerical) and ordinal position in the stack.

Each object is annotated with the following properties:

- `shape` (categorical): shape of the object: cylinder, sphere or cuboid.
- `color` (categorical): 6 colors:
  - Blue. RGB:`[0, 0, 255]`
  - Green. RGB:`[0, 255, 0]`
  - Cyan. RGB:`[0, 255, 255]`
  - Red. RGB:`[255, 0, 0]`
  - Purple. RGB:`[255, 0, 255]`
  - Yellow. RGB:`[255, 255, 0]`

C.2.5 Tetrominoes

This dataset consists of 32×32 images (cropped from the original 35×35 for simplicity) of 3D-textured tetris pieces placed on a black background. There are always 3 objects in a scene, and no occlusions. Objects have different shapes (19 in total), colors (6

fully saturated colors), x and y position.

Each object is annotated with the following properties:

- `shape` (categorical): 19 shapes:
  - Horizontal I piece.
  - Vertical I piece.
  - L piece pointing downward.
  - J piece pointing upward.
  - L piece pointing upward.
  - J piece pointing downward.
  - L piece pointing left.
  - J piece pointing left.
  - J piece pointing right.
  - L piece pointing right.
  - Horizontal Z piece.
  - Horizontal S piece.
  - Vertical Z piece.
  - Vertical S piece.
  - T piece pointing upward.
  - T piece pointing downward.
  - T piece pointing left.
  - T piece pointing right.
  - O piece.
- `color` (categorical): 6 colors:
  - Blue. RGB:`[0, 0, 255]`
  - Green. RGB:`[0, 255, 0]`
  - Cyan. RGB:`[0, 255, 255]`
  - Red. RGB:`[255, 0, 0]`
  - Purple. RGB:`[255, 0, 255]`
  - Yellow. RGB:`[255, 255, 0]`
- `x` (numerical): Horizontal position.

- `y` (numerical): Vertical position.

C.3 Evaluations

In this section, we discuss in more detail the chosen reconstruction and segmentation metrics (Appendix C.3.1), provide implementation details on the downstream property prediction task (Appendix C.3.2), and more closely examine the distribution shifts considered in this study (Appendix C.3.3).

C.3.1 Reconstruction and segmentation metrics

Mean reconstruction error. Since all models in this study are autoencoders, we can use the reconstruction error to This is potentially an informative metric as it should roughly indicate the amount and accuracy of information captured by the models and present in the representations. All models include some form of reconstruction term in their losses, but they may take different forms. We then choose to evaluate the reconstruction error with the mean squared error (MSE), defined for an image $\mathbf{x}$ and its reconstruction $\hat{\mathbf{x}}$ as follows:

$$\text{MSE}\left(\mathbf{x}, \hat{\mathbf{x}}\right) = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 = \frac{1}{D}\sum_{i=1}^{D}(x_i - \hat{x}_i)^2 \tag{C.1}$$

where for simplicity we assume a vector representation of $\mathbf{x}$ and $\hat{\mathbf{x}}$, both with dimension D equal to the number of pixels times the number of color channels.

Adjusted Rand Index (ARI). The Adjusted Rand Index (ARI) (Hubert and Arabi, 1985) measures the similarity between two partitions of a set (or clusterings). Interpreting segmentation as clustering of pixels, the ARI can be used to measure the degree of similarity between two sets of segmentation masks. Segmentation accuracy is then assessed by comparing ground-truth and predicted masks. The expected value of the ARI on random clustering is 0, and the maximum value is 1 (identical clusterings up to label permutation). As in prior work (Burgess et al., 2019; Engelcke et al., 2020; Locatello et al., 2020d), we only consider the ground-truth masks of foreground objects when computing the ARI. Below, we define the Rand Index and the Adjusted Rand Index in more detail.

The Rand Index is a symmetric measure of the similarity between two partitions of a set (Hubert and Arabi, 1985; Rand, 1971; Wagner and Wagner, 2007). It is inspired by traditional classification metrics that compare the number of correctly and incorrectly classified elements. The Rand Index is defined as follows: Let S be a set of n elements, and let $A = \{A_1, \ldots, A_{n_A}\}$ and $B = \{B_1, \ldots, B_{n_B}\}$ be partitions of S. Furthermore, let us introduce the following quantities:

- m_{11}: number of pairs of elements that are in the same subset in both A and B,
- m_{00}: number of pairs of elements that are in different subsets in both A and B,
- m_{10}: number of pairs of elements that are in the same subset in A and in different subsets in B,
- m_{01}: number of pairs of elements that are in different subsets in A and in the same subset in B.

The Rand Index is then given by:

$$\mathrm{RI}(A, B) = \frac{m_{11} + m_{00}}{m_{11} + m_{00} + m_{10} + m_{01}} = \frac{2(m_{11} + m_{00})}{n(n-1)} \tag{C.2}$$

and quantifies the number of elements that have been correctly classified over the total number of elements.

The Rand Index ranges from 0 (no pair classified in the same way under A and B) to 1 (A and B are identical up to a permutation). However, the result is strongly dependent on the number of clusters and on the number of elements in each cluster. If we fix n_A, n_B, and the proportion of elements in each subset of the two partitions, then the Rand Index will increase as n increases, and even converge to 1 in some cases (Fowlkes and Mallows, 1983). The expected value of a random clustering also depends on the number of clusters and on the number of elements n.

The Adjusted Rand Index (ARI) (Hubert and Arabi, 1985) addresses this issue by normalizing the Rand Index such that, with a random clustering, the metric will be 0 in expectation. Given the same conditions as above, let $n_{i,j} = |A_i \cap B_j|$, $a_i = |A_i|$,

and $b_i = |B_i|$, with $i = 1, \ldots, n_A$ and $i = 1, \ldots, n_B$. The ARI is then defined as:

$$\mathrm{ARI}(A, B) = \frac{\sum_{i,j} \binom{n_{i,j}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}} \tag{C.3}$$

which is 0 in expectation for random clusterings, and 1 for perfectly matching partitions (up to a permutation). Note that the ARI can be negative.

Segmentation covering metrics. Segmentation Covering (SC) (Arbelaez et al., 2010) uses the intersection over union (IOU) between pairs of segmentation masks from the sets A and B. How the segmentation masks are matched depends on whether we are considering the covering of B by A (denoted by $A \to B$) or vice versa ($B \to A$). We use the slightly modified definition by Engelcke et al. (2020):

$$\mathrm{SC}(A \to B) = \frac{1}{\sum_{R_B \in B} |R_B|} \sum_{R_B \in B} |R_B| \max_{R_A \in A} \mathrm{IOU}(R_A, R_B) \ , \tag{C.4}$$

where $|R|$ denotes the number of pixels belonging to mask R, and the intersection over union is defined as:

$$\mathrm{IOU}(R_A, R_B) = \frac{|R_A \cap R_B|}{|R_A \cup R_B|} \ . \tag{C.5}$$

While standard (weighted) segmentation covering weights the IOU by the size of the ground truth mask, mean (or unweighted) segmentation covering (mSC) (Engelcke et al., 2020) gives the same importance to masks of different size:

$$\mathrm{mSC}(A \to B) = \frac{1}{|B|} \sum_{R_B \in B} \max_{R_A \in A} \mathrm{IOU}(R_A, R_B) \ , \tag{C.6}$$

where $|B|$ denotes the number of non-empty masks in B. Since a high SC score can still be attained when small objects are not segmented correctly, mSC is considered to be a more meaningful and robust metric across different datasets (Engelcke et al., 2020).

Note that neither SC nor mSC are symmetric: Following Engelcke et al. (2020), we consider A to be the predicted segmentation masks and B the ground-truth masks of the foreground objects. As observed by Engelcke et al. (2020), both SC and mSC penalize over-segmentation (segmenting one object into separate slots), unlike the ARI. Both SC and mSC take values in $[0, 1]$.

C.3.2 Downstream property prediction

Here we start by briefly summarizing the downstream property prediction task presented in the main text, and then provide additional details on the models and evaluation protocol.

Overview of the property prediction task. As outlined in Section 7.3, we evaluate scene representations by training downstream models to predict ground-truth object properties from the representations. Exploiting the fact that object slots share a common representational format, a single downstream model f can be used to predict the properties of each object independently: for each slot representation $\mathbf{z}_k$ we predict a vector of object properties $\hat{\mathbf{y}}_k = f(\mathbf{z}_k)$. This vector represents predictions for *all* properties of an object. We then match each slot's prediction to a corresponding ground-truth object using *mask matching* or *loss matching* (see main text). In non-slotted models such as the VAE baselines considered in this study, we do not have access to separate object representations $\{\mathbf{z}_k\}_{k=1}^K$. Therefore, the downstream model f in this case takes as input the overall distributed representation $\mathbf{z}$, which is a flat vector, and outputs a prediction of *all objects at once*: $\hat{\mathbf{y}} = f(\mathbf{z})$. This is then split into K vectors, which are matched to ground-truth objects with either *loss matching* or *deterministic matching* (see main text).

Implementation details. We use 4 different downstream models: a linear model, and MLPs with up to 3 hidden layers of size 256 each. Let P be the size of the ground-truth property vector, which includes all numerical and categorical[2] properties according to an order specified by the dataset. We denote by K be the number of slots and d the dimensionality of a slot representation $\mathbf{z}_k$ in object-centric models. Note that we must include in $\mathbf{z}_k$ *all* representations related to a slot, possibly including different latent variables that are explicitly responsible for modeling, e.g., the location, appearance, or presence of an object. The downstream model f has input size d and output size P, and is applied in parallel (with shared weights) to all slots. In non-slotted models, we always define the dimensionality of the distributed representation $\mathbf{z}$ in terms of K for fair comparison with slot-based models, hence we can write the latent dimensionality of such models as $d \cdot K$. In this case, the input and output sizes of the downstream model (d and P, respectively) are multiplied by K, and we apply this model only once, to the entire scene representation. The linear downstream

[2]Here we use the one-hot representation of categorical properties.

model is implemented as a linear layer. MLP models (with at least one hidden layer) have hidden size 256 and LeakyReLU nonlinearities, as shown in Table C.8.

Data splits. Let $\mathcal{D}_s$ be a source dataset and $\mathcal{D}_t$ a target dataset. When doing in-distribution evaluation, we train and test the downstream model without distribution shifts, so we simply have $\mathcal{D}_s = \mathcal{D}_t$. Given a representation function r, and a matching strategy to match the slots with ground-truth objects, we consider:

- a train split of 10 000 images from $\mathcal{D}_s$,
- a validation split of 1000 images from $\mathcal{D}_s$,
- a test split of 2000 images from $\mathcal{D}_t$.

The test split only contains images that were not used when training the upstream unsupervised models.

Training. We then train the downstream model to predict $\hat{\mathbf{y}}$ from $\mathbf{z} = r(\mathbf{x})$ using the Adam optimizer with an initial learning rate of 1e-3 and a batch size of 64, for a maximum of 6000 steps. The learning rate is halved every 2000 steps. We perform early stopping as follows: We use the validation set to compute the (in-distribution) validation loss every 250 training steps—if the loss does not decrease by more than 0.01 for 3 evaluations (750 steps), training is interrupted. In this stage, the representation for each image is fixed, i.e. the representation function r is never updated. The loss is computed independently for each object property, and is a sum of MSE and cross-entropy terms, depending on whether an object property is numerical or categorical.

Table C.8. Architecture of the downstream MLP models for property prediction. The third and fourth items are repeated 0 or more times, depending on the required number of hidden layers.

Layer type	Input size	Output size	
Linear	d or $d \cdot K$	256	
LeakyReLU(0.01)	256	256	
Linear	256	256	} repeated 0 or
LeakyReLU(0.01)	256	256	} more times
Linear	256	P or $P \cdot K$	

Downstream training and evaluation under distribution shifts. As mentioned earlier, when doing in-distribution evaluation we simply have $\mathcal{D}_s = \mathcal{D}_t$. In the general case, we may for example train on the original Multi-dSprites dataset, and test on the Multi-dSprites variant that has an unseen shape or an occlusion. In the special case in which we allow retraining of the downstream model (see Sections 7.4.3 and 7.4.4), we still have $\mathcal{D}_s = \mathcal{D}_t$, but they are both OOD with respect to the original "clean" dataset used for training the unsupervised models.

Under distribution shifts, the representations $r(\mathbf{x})$ might be inaccurate, which might bias our downstream results. Although there is no perfect solution to this issue, we attempt to reduce as much as possible the potential effect of distribution shifts on the training and evaluation of downstream models. When distribution shifts affect global scene properties, there is no alternative but to train and evaluate the models as usual. When distribution shifts affect single objects, however, we can assume that the representations of the ID objects are not as severely affected by the shift, and only use these for training downstream models.

Here we consider the case where the test dataset $\mathcal{D}_t$ has an object-level distribution shift, and the training dataset $\mathcal{D}_s$ is either the original "clean" dataset or the same as $\mathcal{D}_t$. At **train time**, we ignore OOD objects (if any) both when matching slots with objects and when training downstream property prediction models. Note that, when the training dataset $\mathcal{D}_s$ is the original "clean" dataset, the downstream models are always trained as usual because there are no OOD objects. At **test time**, there are a few cases depending on the matching strategy:

- When using mask matching, we consider *all* objects for matching, and evaluate the downstream models on all objects. We then report test results on ID and OOD objects separately.

- When using loss matching, we cannot match all ground-truth objects, since the OOD objects might have OOD categorical properties (in our setup, the downstream models cannot predict classes that were not seen during training). Therefore, we resort to a two-step matching approach: we first match slots to all objects using the prediction loss computed only on the properties that are ID for *all* objects. We then keep only the matches for OOD objects, and repeat the usual loss matching with the remaining slots and objects, using all properties. The OOD objects are thus matched in a relatively fair way, while the matching of the ID objects can be refined at a later step using all available properties.

- When using deterministic matching, we cannot exactly follow the two-step matching strategy presented above. Instead, we modify the lexicographic order to give a higher weight to OOD features of OOD objects, so the corresponding objects are pushed down in the order while maintaining the order given by more significant (according to the order) properties. Note that the downstream model in this case might be at a disadvantage if it is trained on a dataset with object-level distribution shifts: the model is now trained to predict only ID objects, so at test time there will be one more target object on average.

C.3.3 Distribution shifts for OOD evaluation

Here we present more in detail the distribution shifts we apply to images in order to test OOD generalization in different scenarios. Examples are shown in Fig. C.2.

Occlusion. A gray square is placed on top of the scene. The position is determined by picking 5 locations uniformly at random (such that the entire square is in the im-

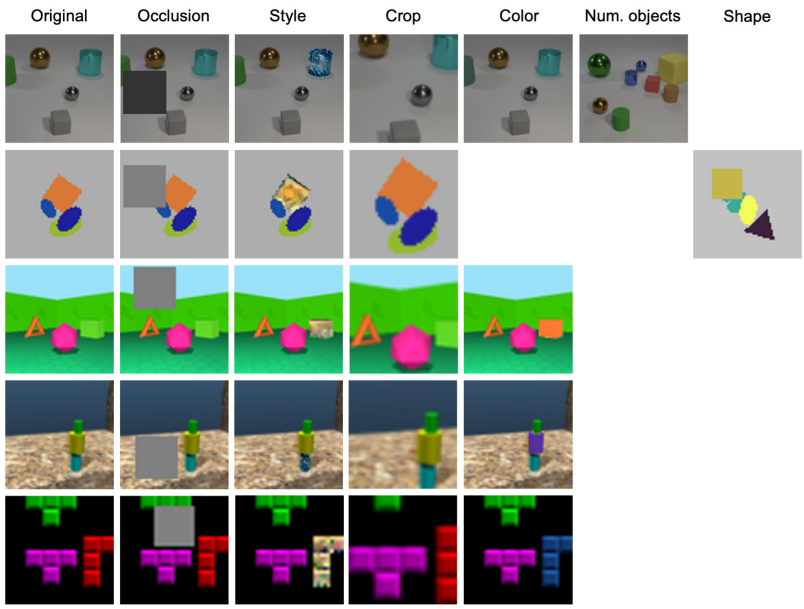


Figure C.2. Distribution shifts applied to the different datasets to test generalization.

age) and selecting the one that occludes less (in terms of total area) of the foreground objects. The size of the occlusion is $(\lfloor 0.4 \cdot H \rfloor, \lfloor 0.4 \cdot W \rfloor)$ with H and W the height and width of the image, respectively. Occluded objects have their mask updated to reflect the occlusion. The occlusion is categorized as background (or first background object in case there are multiple background objects such as in Objects Room). The RGB color of the square is $[0.2, 0.2, 0.2]$ for CLEVR and $[0.5, 0.5, 0.5]$ for all other datasets.

Object color. An object is selected uniformly at random and its color is changed by randomly adjusting its brightness, contrast, saturation, and hue, using torchvision's `ColorJitter` transform with arguments $[0.5, 0.5, 0.5, 0.5]$ for the four above-mentioned parameters. This transformation is not performed on Multi-dSprites, since the object colors in this dataset cover the entire RGB color space. The `color` and `material` properties (when relevant) are not used in downstream tasks.

Crop. The image and mask are cropped at the center and resized to match their original size. The crop size is $(\lfloor \frac{2}{3} H \rfloor, \lfloor \frac{2}{3} W \rfloor)$ with H and W the original height and width of the image, respectively. When resizing, we use bilinear interpolation for the image and nearest neighbor for the mask.

Object style. We implement style transfer based on Gatys, Ecker, and Bethge (2016) and on the PyTorch tutorial by Jacq and Herring (2021). The first 100k samples in all datasets are converted using as style image *The Great Wave off Kanagawa* from Hokusai's series *Thirty-six Views of Mount Fuji*. The style is applied only to one foreground object using the object masks. The `color` and `material` properties (when relevant) are not used in downstream tasks.

Object shape. For the Multi-dSprites dataset, a triangle is placed on the scene with properties sampled according to the same distributions defined by the Multi-dSprites dataset. This is performed only on the images where at most 4 objects are present, to mimic changing the shape of an existing object. The depth of the triangle in the object stack is selected uniformly at random as an integer in $[1, 5]$. All objects from the selected depth and upwards are moved up by one level to place the new shape underneath them. The objects masks are adjusted accordingly for both the added shape and the objects below it.

C.4 Additional results

In this section, we report additional quantitative results and show qualitative performance on all datasets for a selection of object-centric models and VAE baselines.

C.4.1 Performance in the training distribution

Fig. C.3 shows the distributions of the reconstruction MSE and all the segmentation metrics, broken down by dataset and model. The relationship between these metrics is also shown in scatter plots in Fig. C.4. As discussed in Section 7.4.1, we observe that segmentation covering metrics are correlated with the ARI only in some cases, and the models are ranked very differently depending on the chosen segmentation metric. In particular, we observe here that Slot Attention achieves a high ARI score and significantly lower (m)SC scores on CLEVR, Multi-dSprites, and Tetrominoes. This is because Slot Attention on these datasets tends to model the background across many slots (see Appendix C.4.3), which is penalized by the denominator of the IOU in the (m)SC scores (see (C.4) to (C.6) in Appendix C.3.1). This behavior should not have a major effect on downstream performance, which is confirmed by the strong and consistent correlation between ARI and downstream performance (see also Section 7.4.2 and Fig. C.7).

Fig. C.5 shows an overview of downstream factor prediction performance on all labeled datasets (one per column), using as downstream predictors a linear model or an MLP with up to 3 hidden layers (one model per row). The MLP1 results are also shown in Section 7.4.2 (Fig. 7.4). We report results separately for each object-centric model and for each ground-truth object property. The metrics used here are accuracy for categorical attributes and R^2 for numerical attributes. We generally observe consistent trends across downstream models. In Fig. C.6, we show the same results in a different way, to directly compare downstream models (here the median baselines for slot-based and distributed representations are shown as horizontal lines on top of the relevant bars). Using larger downstream models tends to slightly improve test performance but, interestingly, in many cases the effect is negligible. There are however a few cases in which using a larger model significantly boosts test performance in object property prediction. In some cases it seems sufficient to use a small MLP with one hidden layer instead of a linear model (e.g., color prediction in CLEVR with Slot Attention, shape prediction in CLEVR with MONet and Slot

Attention, color prediction in Tetrominoes with MONet, GENESIS, and SPACE, or location prediction in Multi-dSprites with SPACE), while in other cases we get further gains by using even larger models (e.g., shape prediction in Multi-dSprites with SPACE, and shape prediction in Tetrominoes with all models except Slot Attention which already achieves a perfect score with a linear model). Results for VAEs are generally less interpretable because the performance is often too close to the naive baseline. However, in some cases using deeper downstream models has clear benefits: e.g., shape prediction in Tetrominoes and color prediction in Shapestacks improve from baseline level when using a linear model to a relatively high accuracy when using one or two hidden layers. In other cases, a linear model already works relatively well even from distributed representations—although significantly worse than object-centric representations—and using deeper downstream models is not beneficial (e.g., color and size prediction in CLEVR). Finally, in many other cases, larger downstream models do not seem to be sufficient to improve performance from VAE representations, confirming that often the relevant information may not be easily accessible and suggesting that object-centric representations may be generally beneficial.

In Fig. C.7 we show the Spearman rank correlations between evaluation metrics and downstream performance with all considered combinations of slot matching (loss- and mask-based) and downstream model (linear, MLP with 1, 2, or 3 hidden layers). The trends are broadly consistent in all combinations, except that correlations with ARI tend to be stronger (perhaps unsurprisingly) when using mask matching, and when using larger downstream models.

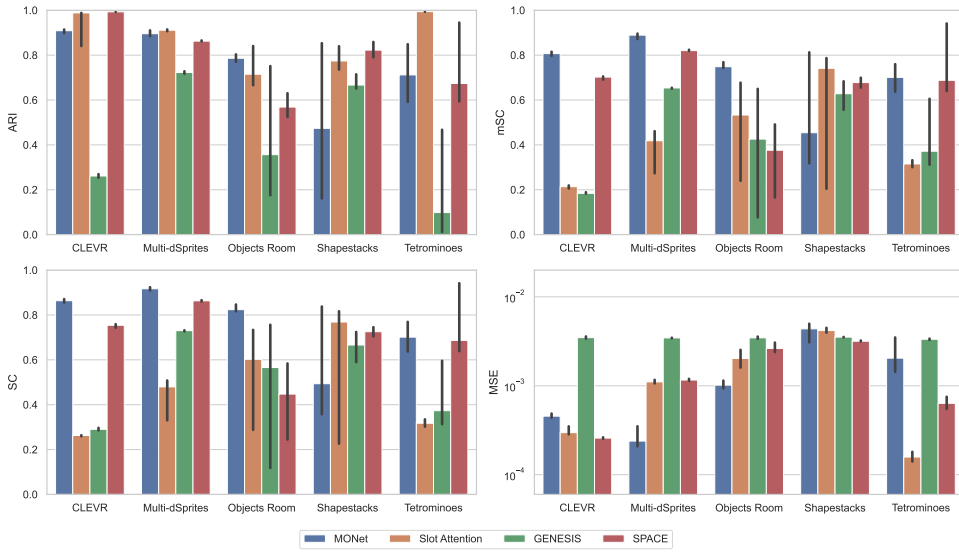


Figure C.3. Overview of segmentation metrics (ARI ↑, mSC ↑, SC ↑) and reconstruction MSE (↓) in distribution (test set of 2000 images). The bars show medians and 95% confidence intervals with 10 random seeds.

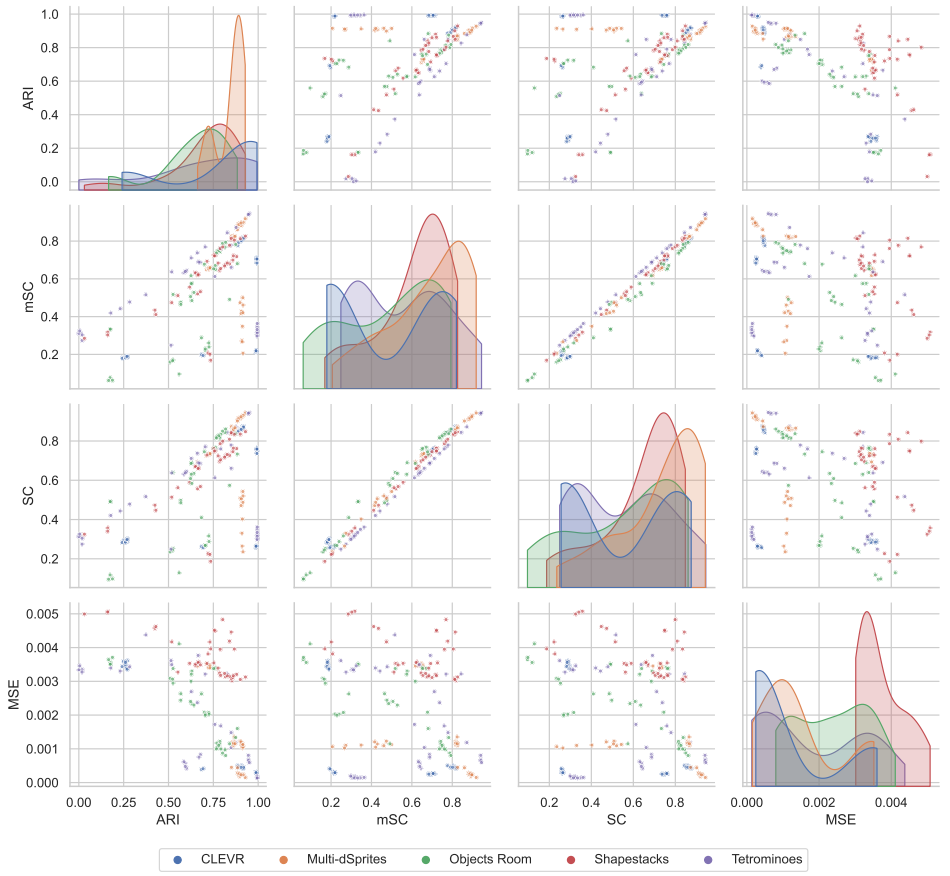


Figure C.4. Scatter plots between metrics over all 200 object-centric models, color-coded by dataset. Diagonal plots: kernel density estimation (KDE) of the quantities on the x-axes.

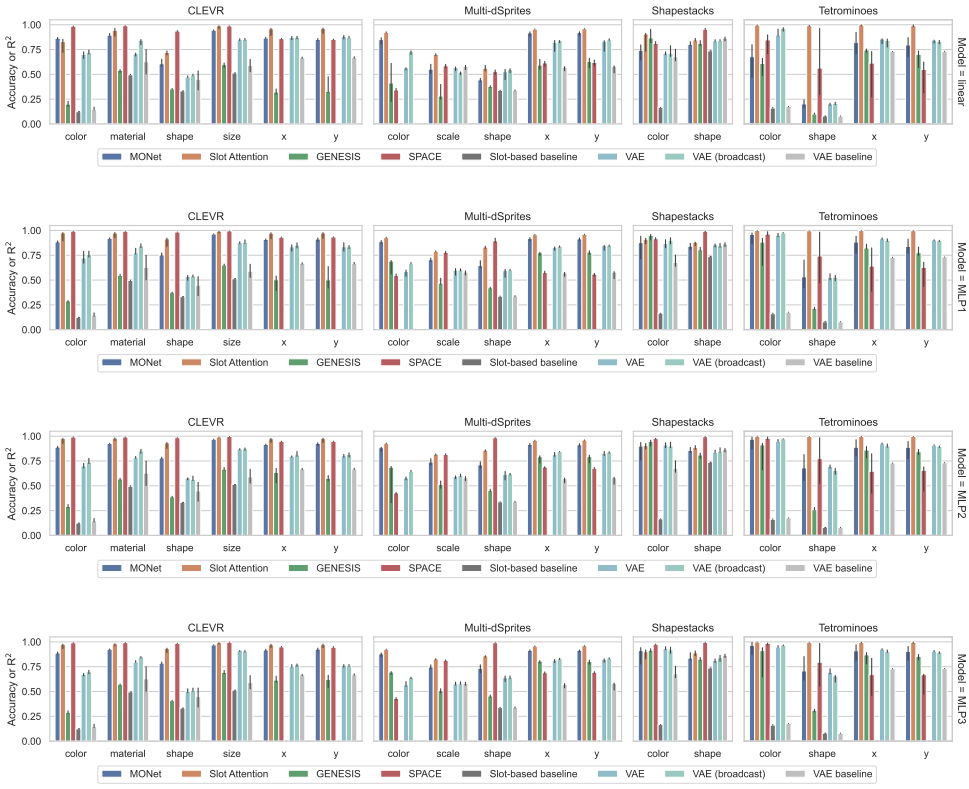


Figure C.5. Overview of downstream performance in the training distribution (test set of 2000 images) for object-centric models and VAEs, with respective baselines. The metrics on the y-axes are accuracy (↑) for categorical properties and R^2 (↑) for numerical features. Each row show results for a different downstream prediction model. From top to bottom: linear, MLP with 1, 2, and 3 hidden layers (see annotation on the right). We use loss matching (see Section 7.3) for all models. The bars show medians and 95% confidence intervals with 10 random seeds.

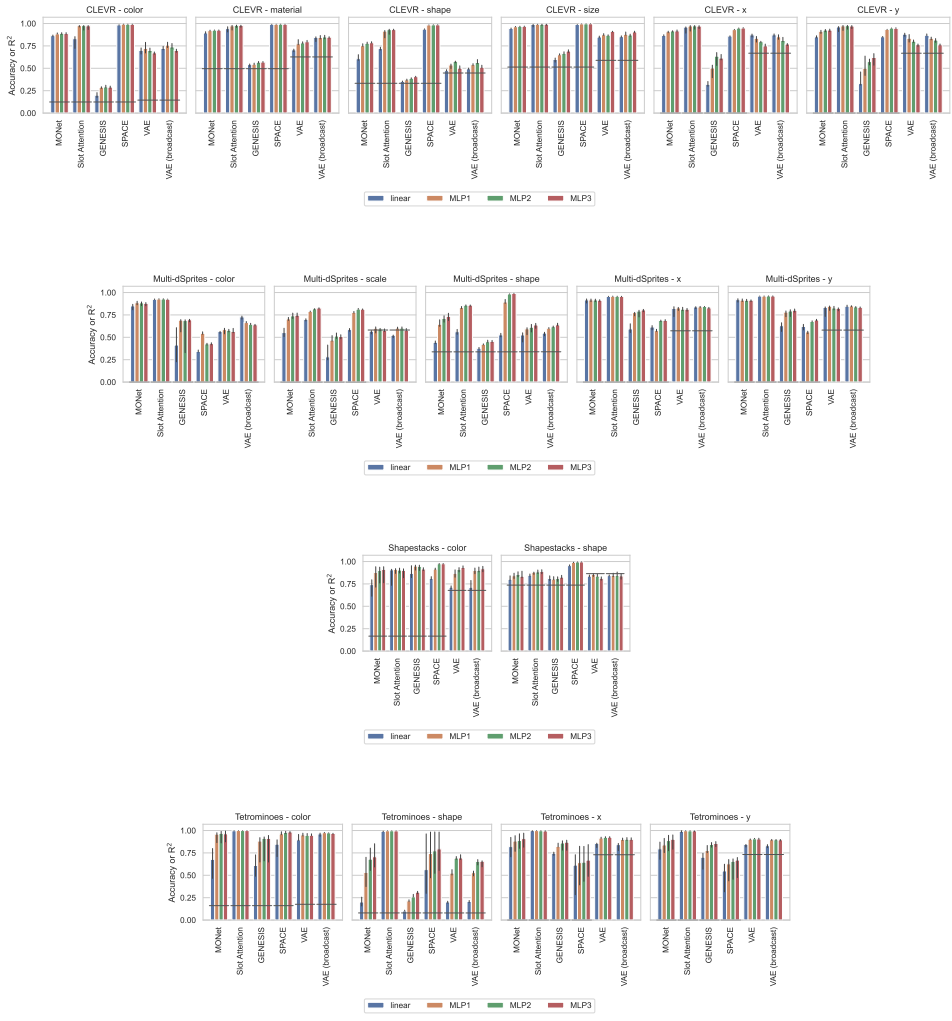


Figure C.6. Comparing property prediction performance of different downstream models (linear, MLP with 1 to 3 hidden layers), using loss matching (see Section 7.3). Results on a test set of 2000 images in the training distribution of the upstream unsupervised models. Each plot shows the test performance on one feature of a dataset. We show results for all object-centric models and VAEs, and indicate the baseline (see Section 7.3) with a horizontal line (not visible when the baseline is 0). The metrics on the y-axes are accuracy (↑) for categorical properties and R^2 (↑) for numerical features. The bars show medians and 95% confidence intervals with 10 random seeds.

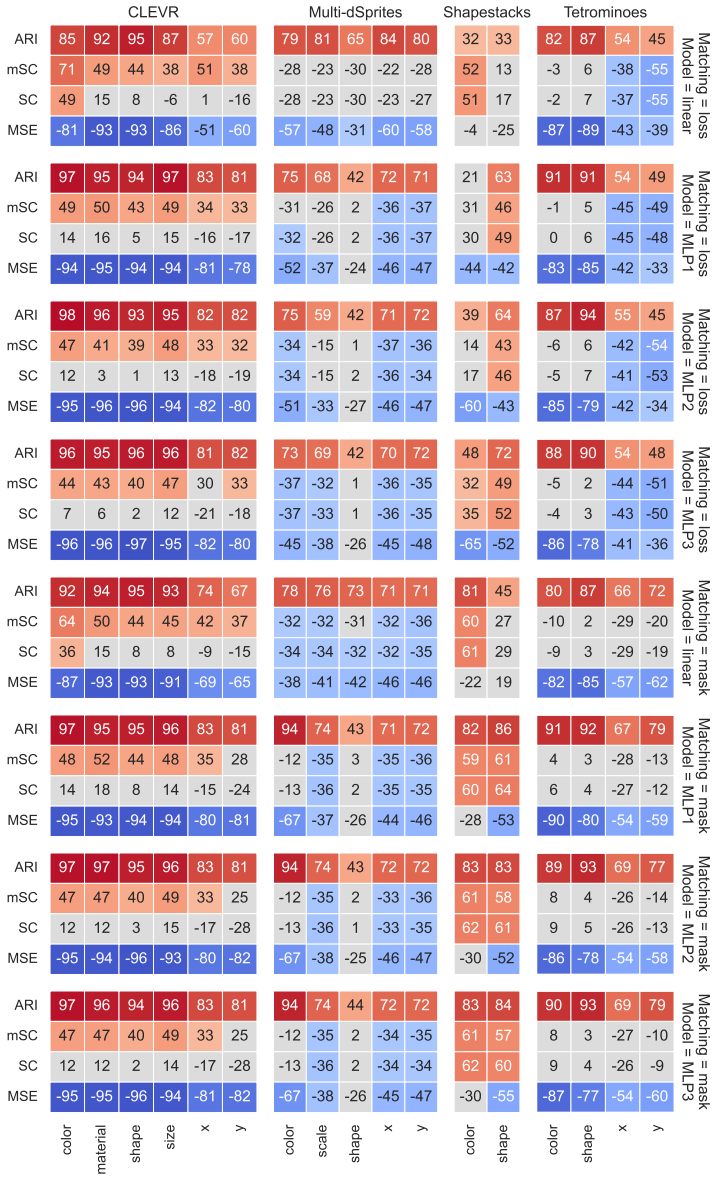


Figure C.7. Spearman rank correlations between evaluation metrics and downstream performance with all considered combinations of slot matching (loss- and mask-based) and downstream model (linear, MLP with 1, 2, or 3 hidden layers). The correlations are color-coded only when $p<0.05$.

C.4.2 Performance under distribution shifts

C.4.2.1 Segmentation and reconstruction

In Fig. C.8, we report the distributions of the reconstruction MSE and segmentation metrics in scenarios where one object is OOD. Results are split by dataset, model, and type of distribution shift. As discussed in Section 7.4.3, the SC and mSC scores show a compatible but less pronounced trend, while the MSE more closely mirrors ARI. Notably, in many cases when we alter object style or color, the reconstruction MSE increases significantly while the ARI is only mildly affected. This suggests that the models are still capable of separating the objects but, unsurprisingly, they fail at reconstructing them properly as they have features that were never encountered during training.

Fig. C.9 shows analogous results when the distribution shift affects global scene properties. Here we observe that segmentation performance is relatively robust to occlusion although the MSE increases significantly (as expected, the occlusion cannot be reconstructed properly). Segmentation metrics are also robust to increasing the number of objects in CLEVR—here the MSE also increases, but to a lesser extent, especially for SPACE.

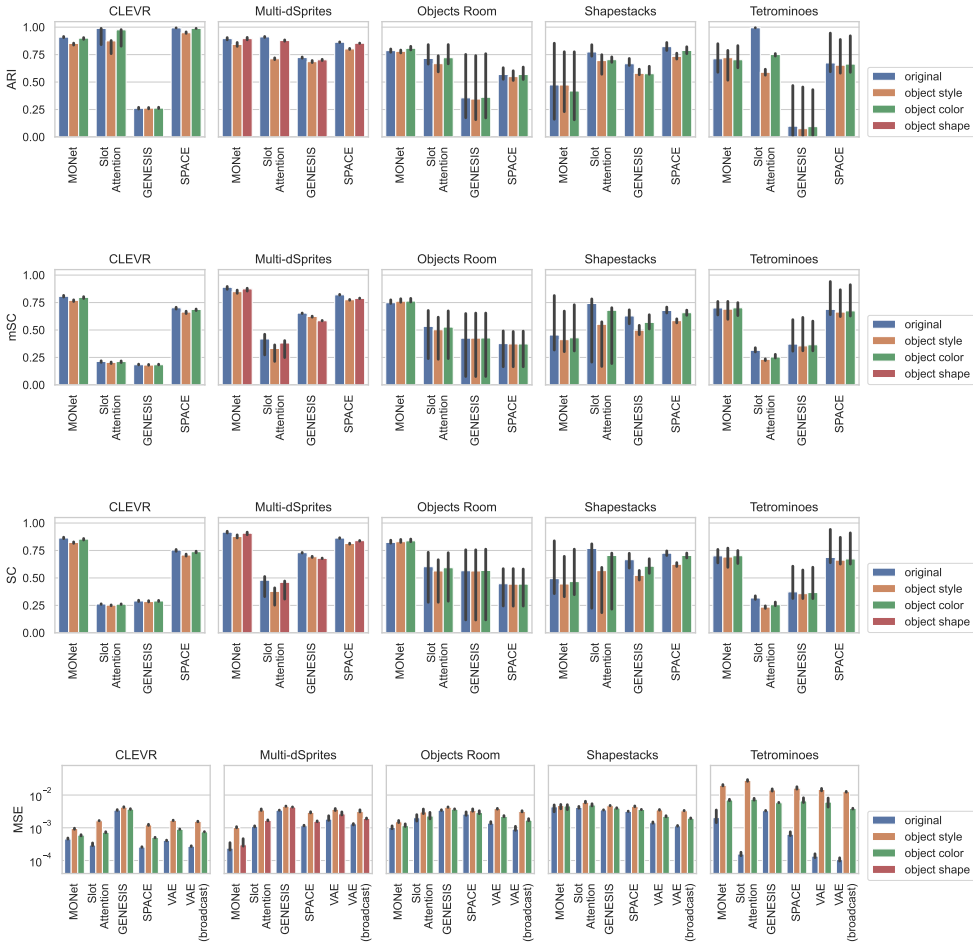


Figure C.8. Overview of segmentation metrics (ARI ↑, mSC ↑, SC ↑) and reconstruction MSE (↓) on OOD dataset variants where **one object** undergoes distribution shift (test set of 2000 images). The bars show medians and 95% confidence intervals with 10 random seeds.

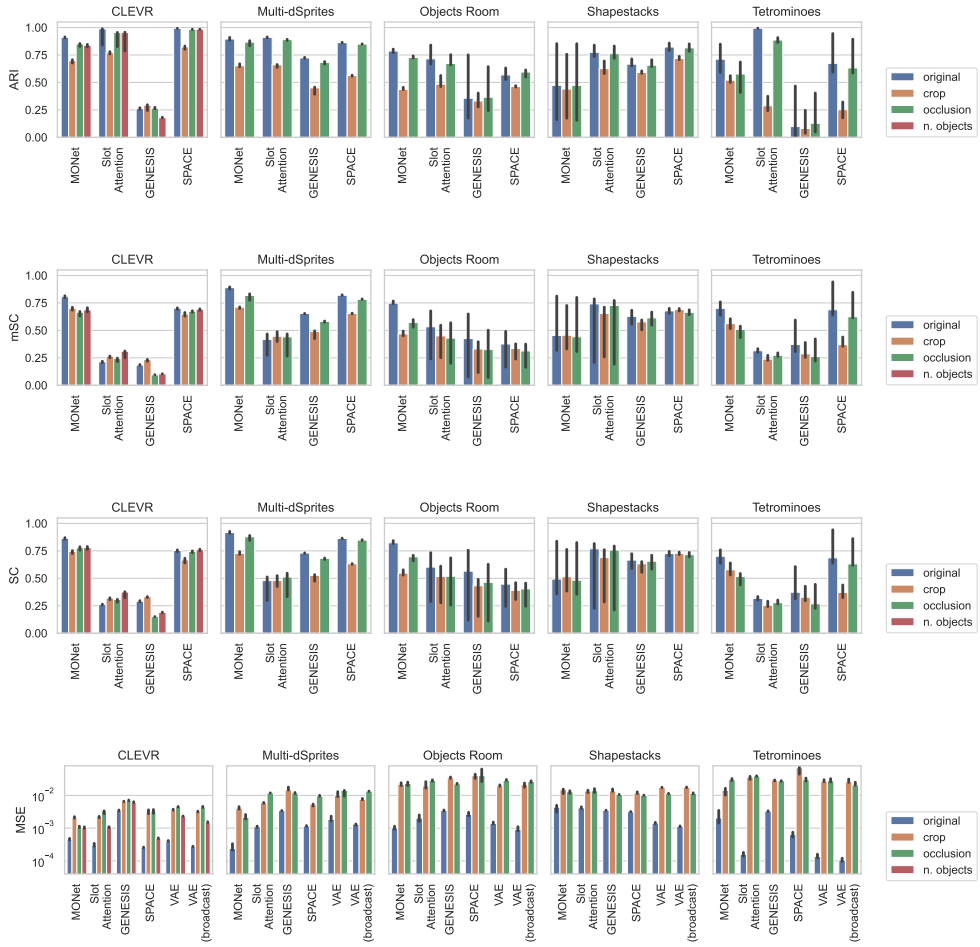


Figure C.9. Overview of segmentation metrics (ARI ↑, mSC ↑, SC ↑) and reconstruction MSE (↓) on OOD dataset variants where **global properties** of the scene are altered (test set of 2000 images). The bars show medians and 95% confidence intervals with 10 random seeds.

C.4.2.2 Downstream performance

In Figs. C.10 to C.25, we show the relationship between ID and OOD downstream prediction performance for the same model, dataset, downstream predictor, and object property. Assume a pretrained unsupervised object discovery model is given, and a downstream model is trained from said model's representations to predict object properties. These plots answer the following question: given that the downstream model predicts a particular object property (e.g., size in CLEVR) with a certain accuracy (on average over all objects in all test images), how well is it going to predict the same property when the scene undergoes one of the possible distribution shifts considered in this study? And in case the distribution shift only affects one object, how well is it going to predict that property in the ID objects as opposed to the OOD objects? These 16 figures show all combinations of the following 4 factors (hierarchically in this order): object-centric/distributed representations; loss/mask matching for object-centric representations or loss/deterministic for distributed representations; without/with retraining of the downstream model after the distribution shift has occurred; single-object/global distribution shifts. In each figure, we show results for each of the 4 downstream models considered in this study (linear, and MLP with up to 3 hidden layers). For each of these, we show splits in terms of ID/OOD objects (when applicable), dataset, upstream model, type of distribution shift.

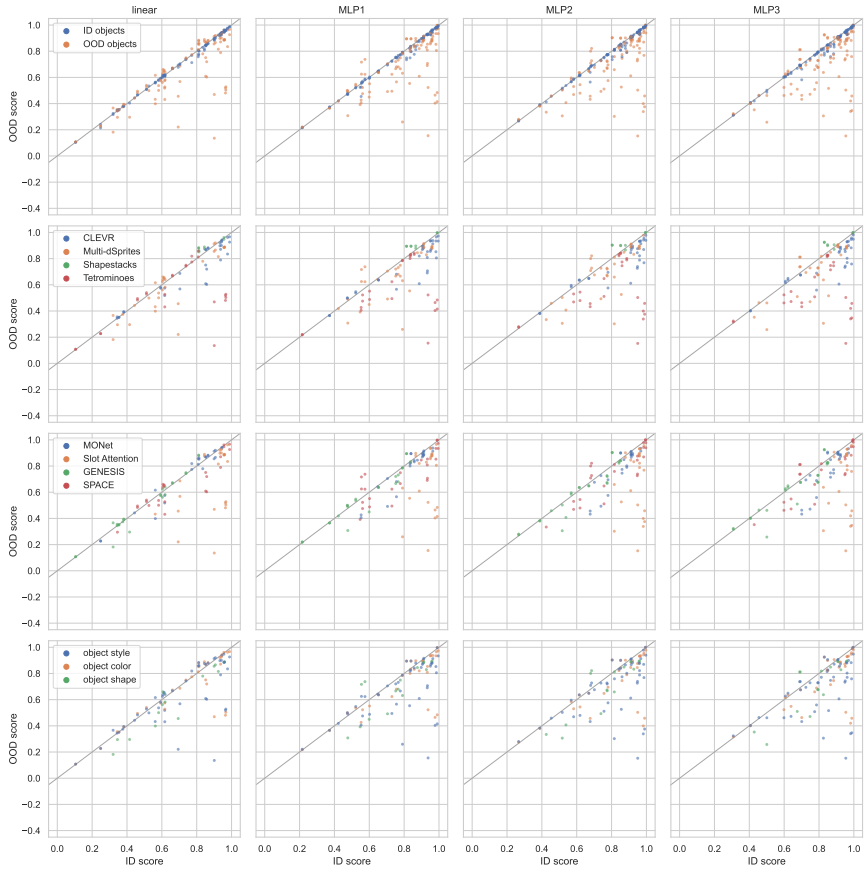


Figure C.10. Generalization of **object-centric representations** in downstream prediction, using **loss matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

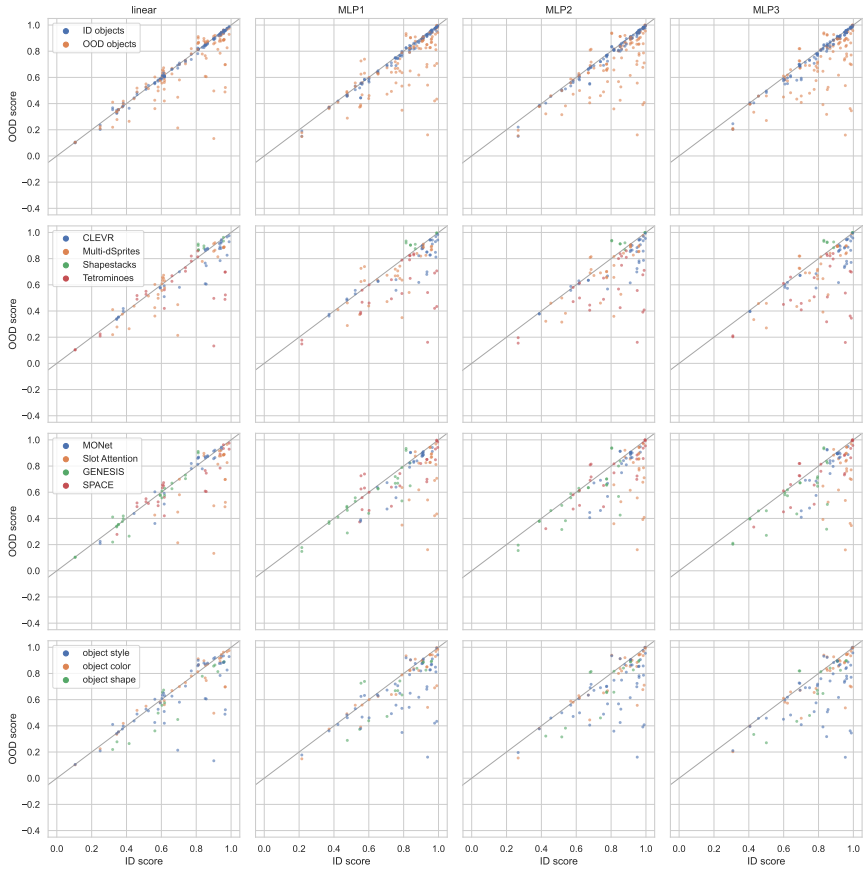


Figure C.11. Generalization of **object-centric representations** in downstream prediction, using **loss matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

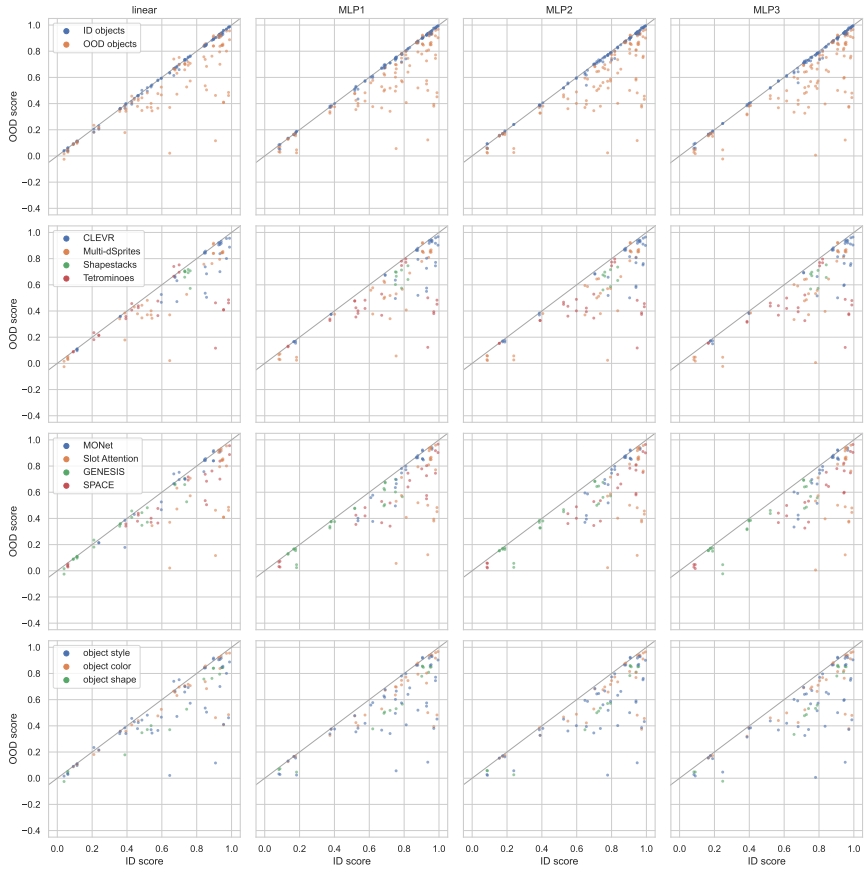


Figure C.12. Generalization of **object-centric representations** in downstream prediction, using **mask matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

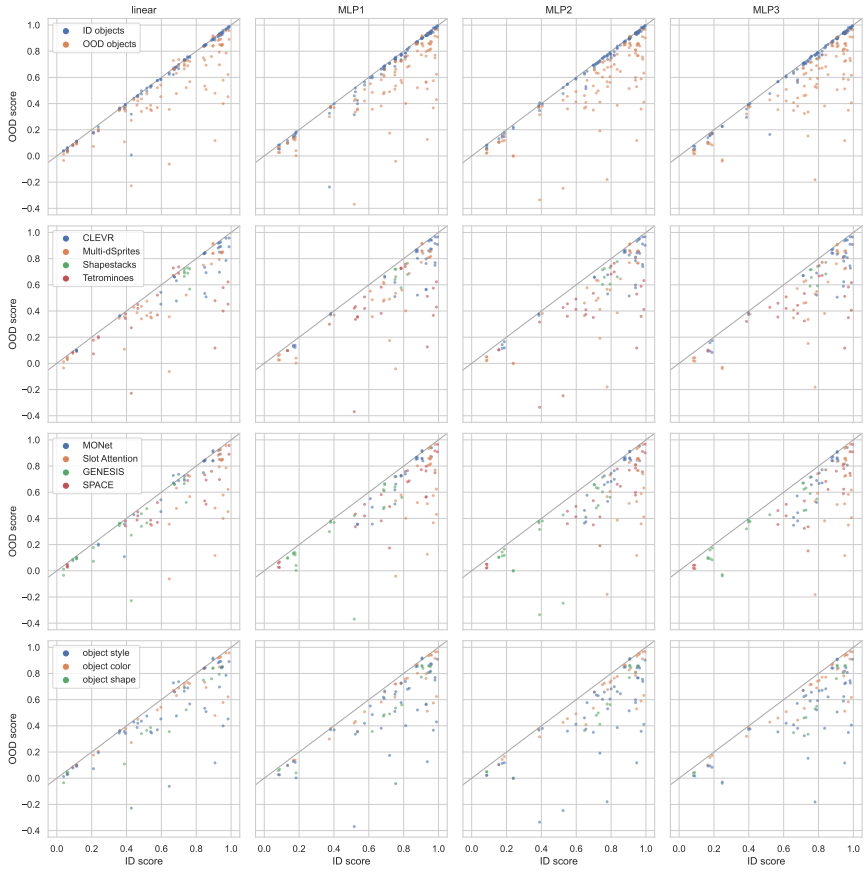


Figure C.13. Generalization of **object-centric representations** in downstream prediction, using **mask matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

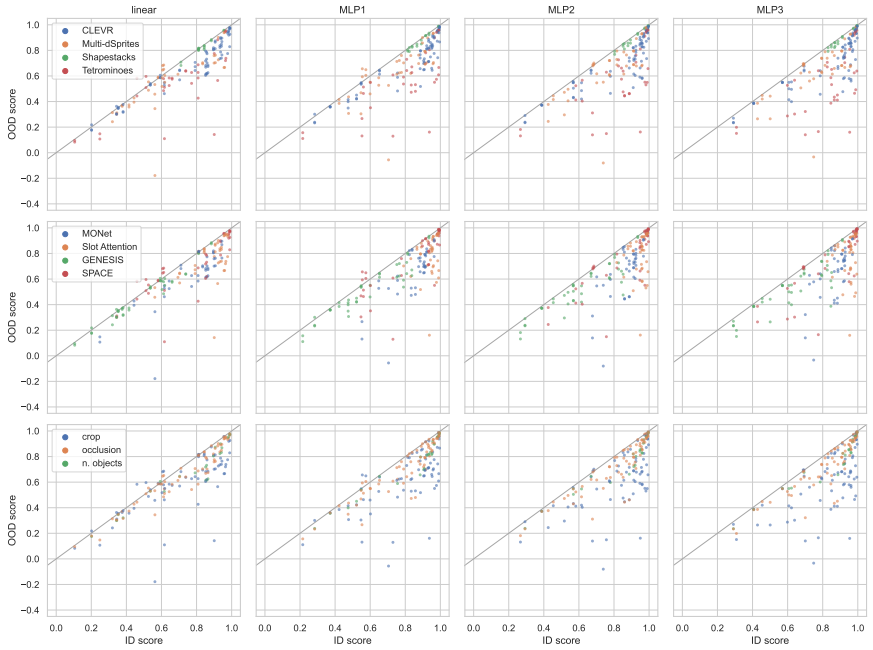


Figure C.14. Generalization of **object-centric representations** in downstream prediction, using **loss matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

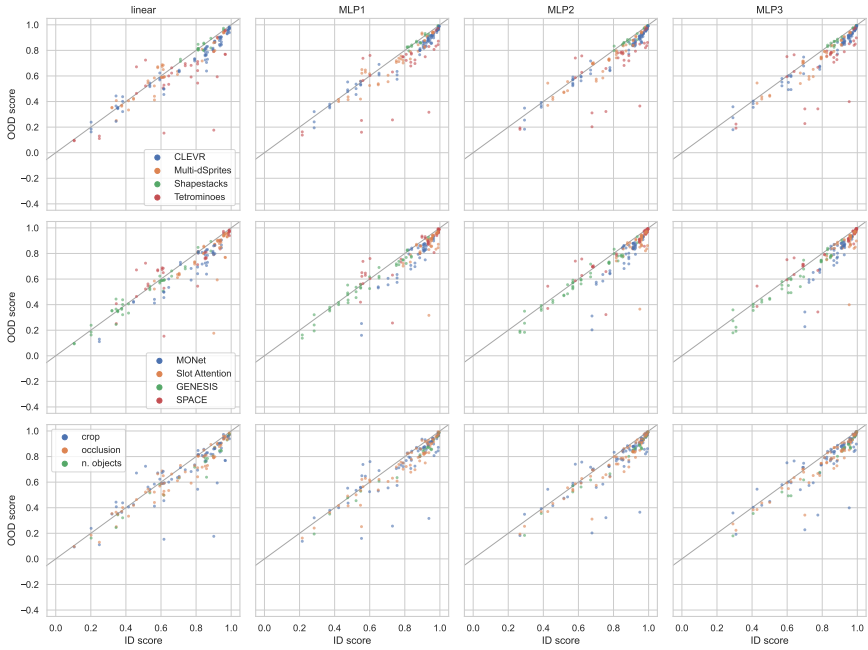


Figure C.15. Generalization of **object-centric representations** in downstream prediction, using **loss matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

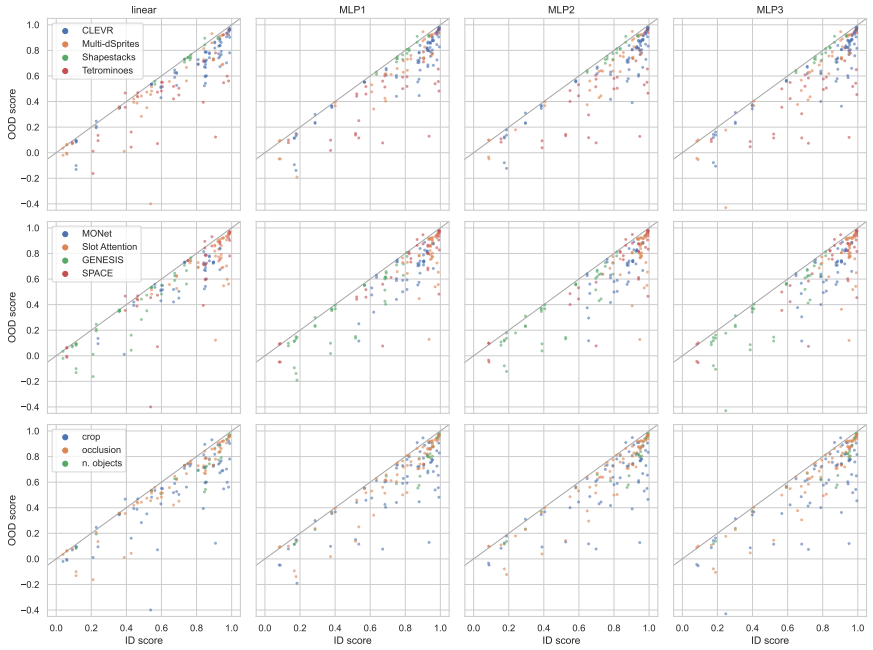


Figure C.16. Generalization of **object-centric representations** in downstream prediction, using **mask matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

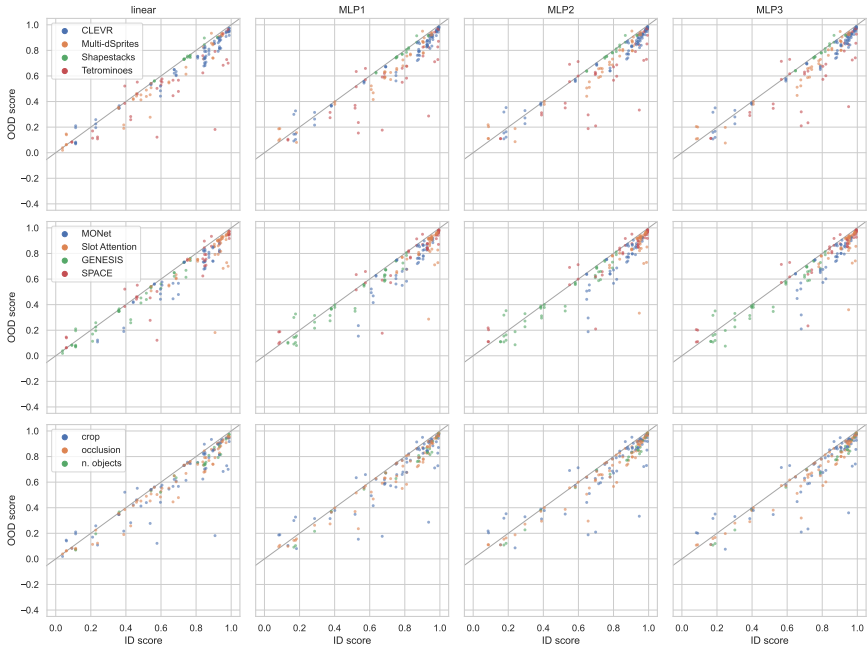


Figure C.17. Generalization of **object-centric representations** in downstream prediction, using **mask matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

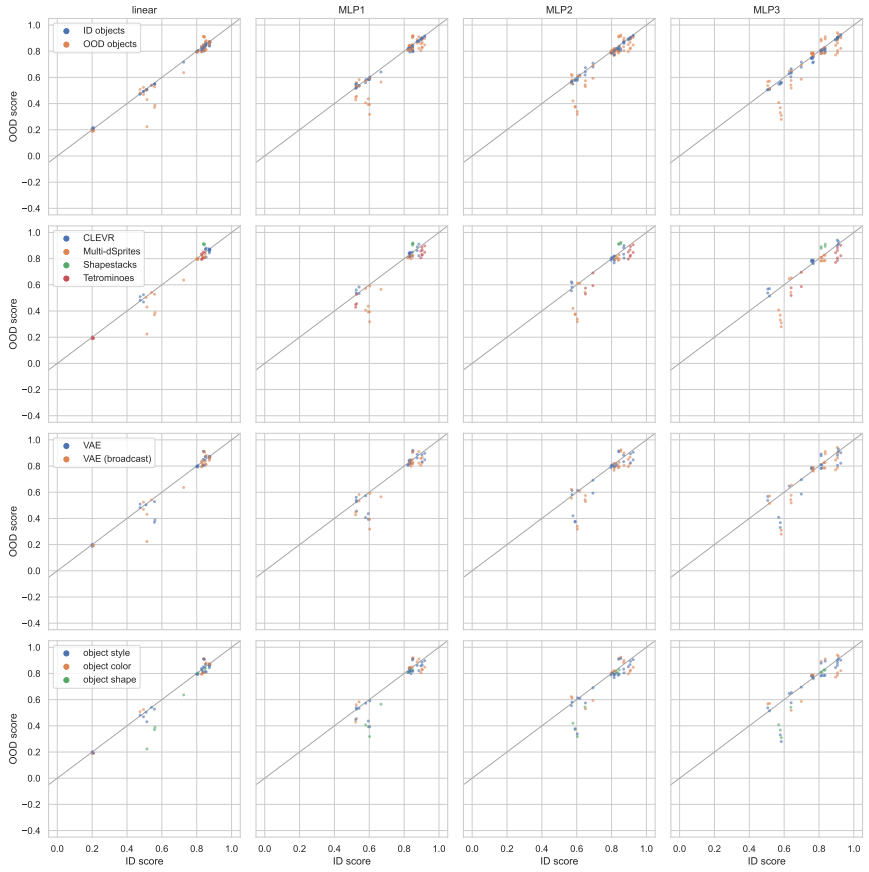


Figure C.18. Generalization of **distributed representations** in downstream prediction, using **loss matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

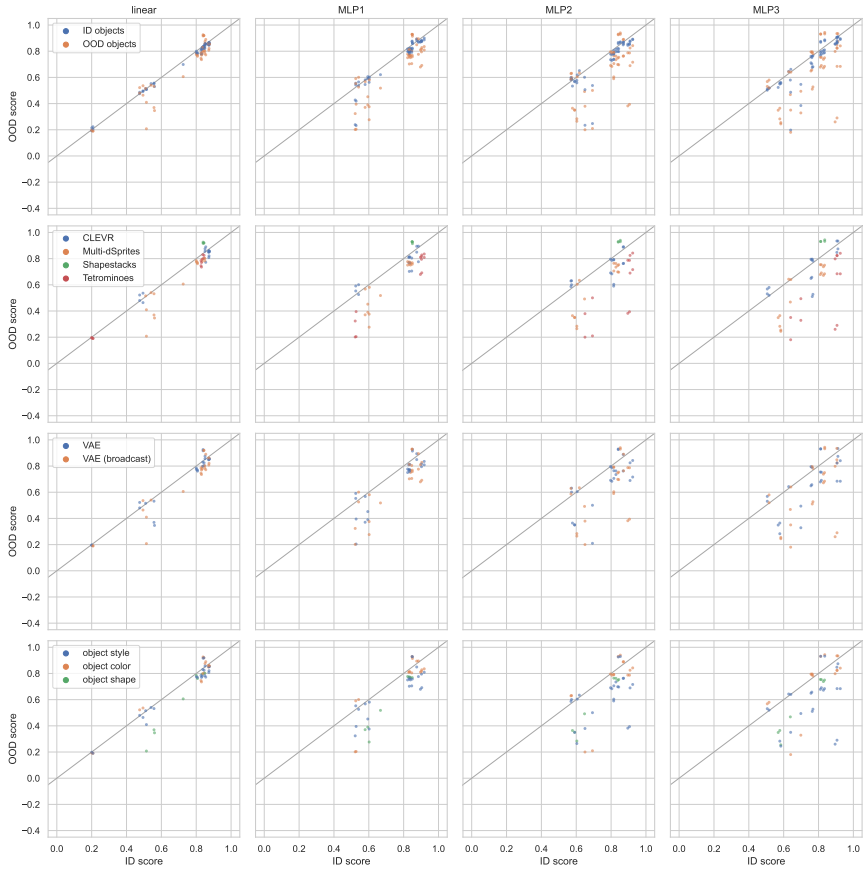


Figure C.19. Generalization of **distributed representations** in downstream prediction, using **loss matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

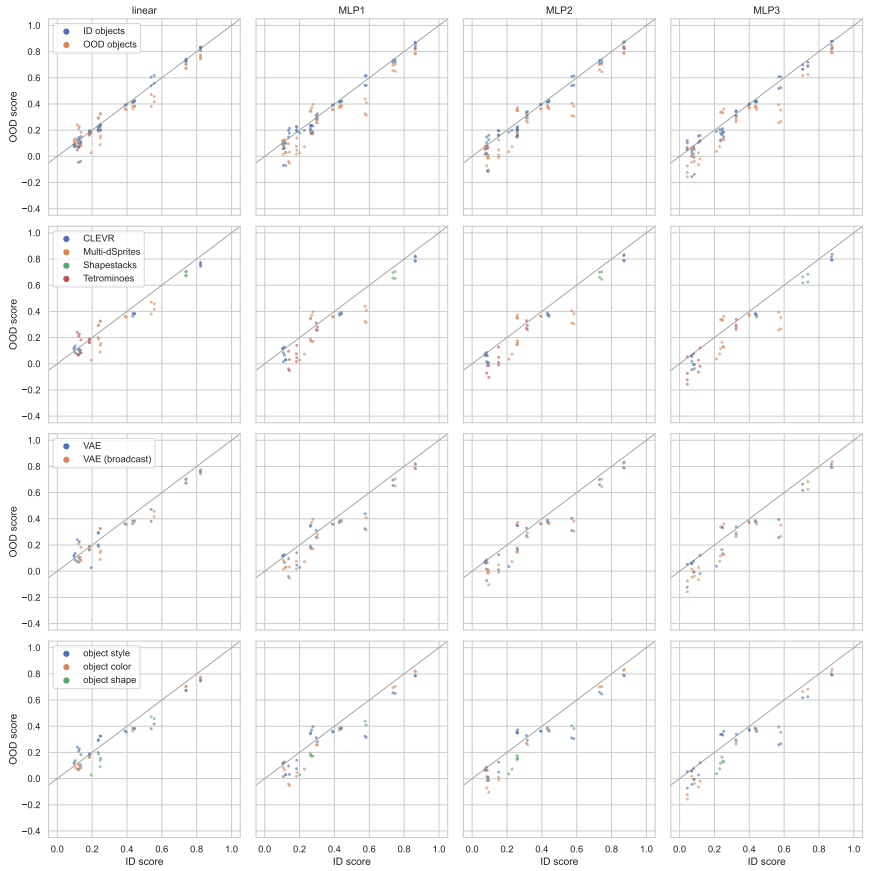


Figure C.20. Generalization of **distributed representations** in downstream prediction, using **deterministic matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

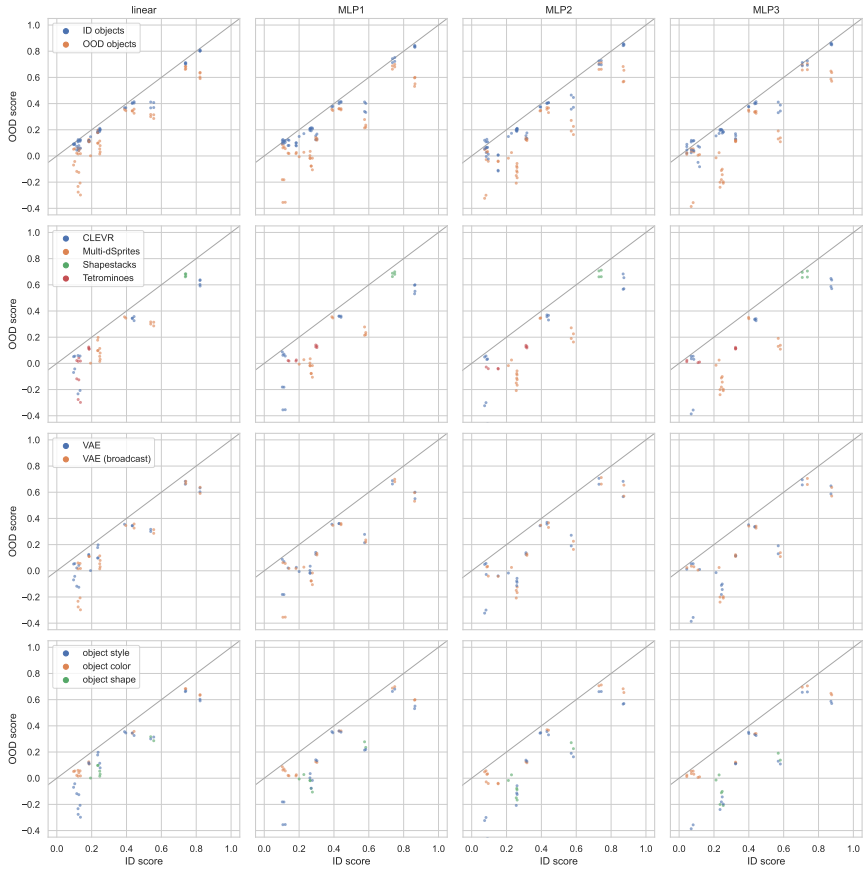


Figure C.21. Generalization of **distributed representations** in downstream prediction, using **deterministic matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **one object**. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, one type of distribution shift, and either ID or OOD objects. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts and to ID/OOD objects. In the top row, we separately report (color-coded) the performance over ID and OOD objects. In the following rows, we only show OOD objects and split according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

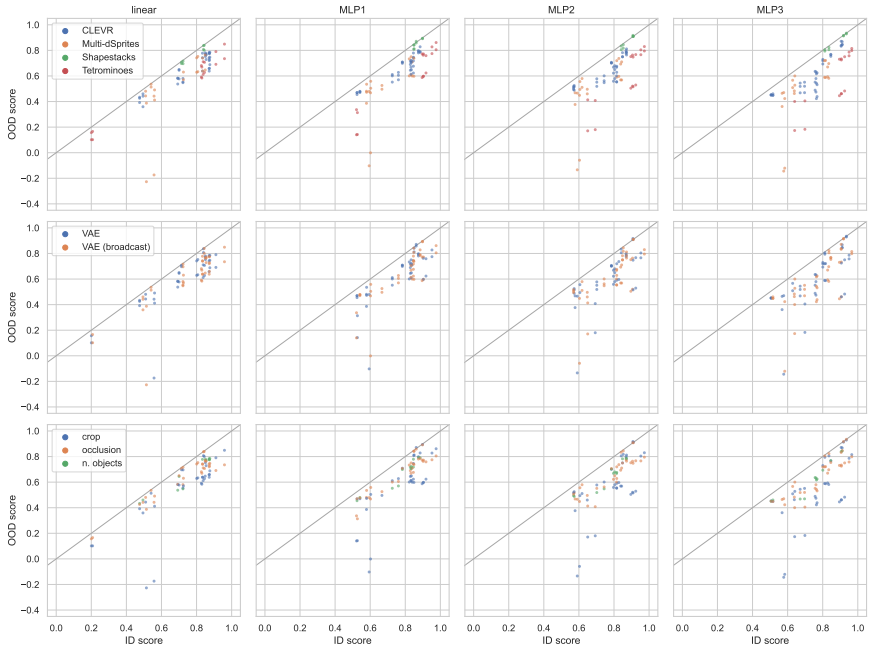


Figure C.22. Generalization of **distributed representations** in downstream prediction, using **loss matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

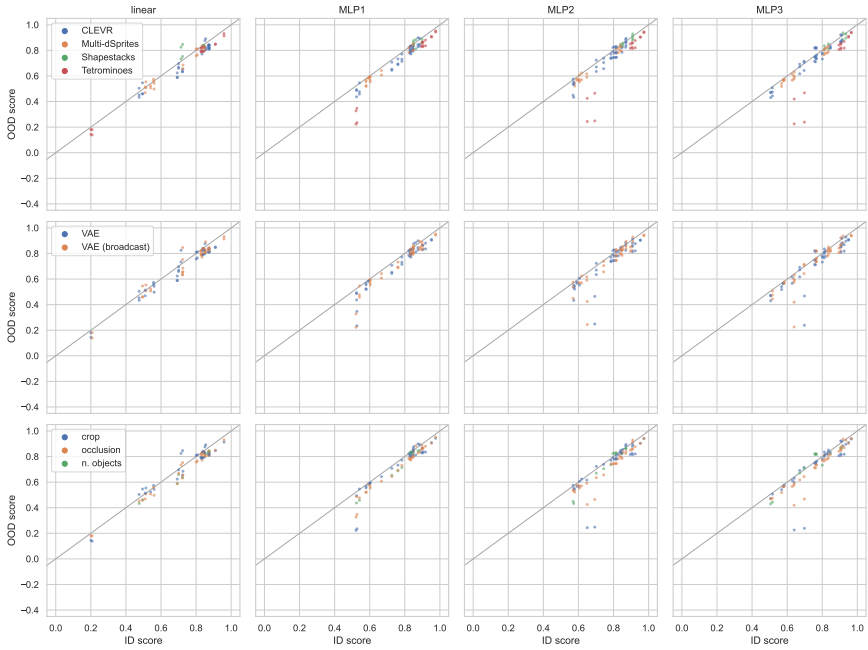


Figure C.23. Generalization of **distributed representations** in downstream prediction, using **loss matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

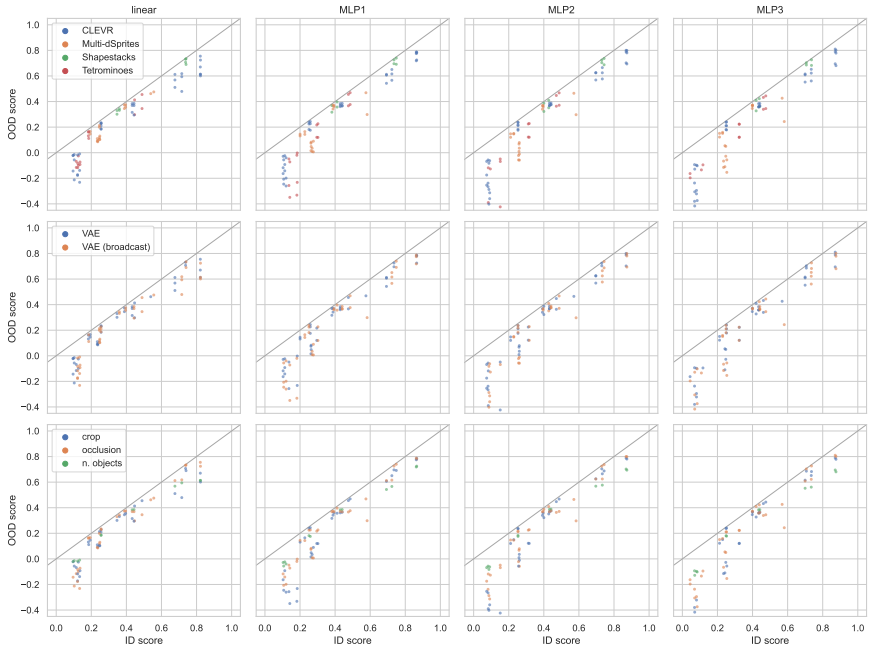


Figure C.24. Generalization of **distributed representations** in downstream prediction, using **deterministic matching** and **without retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

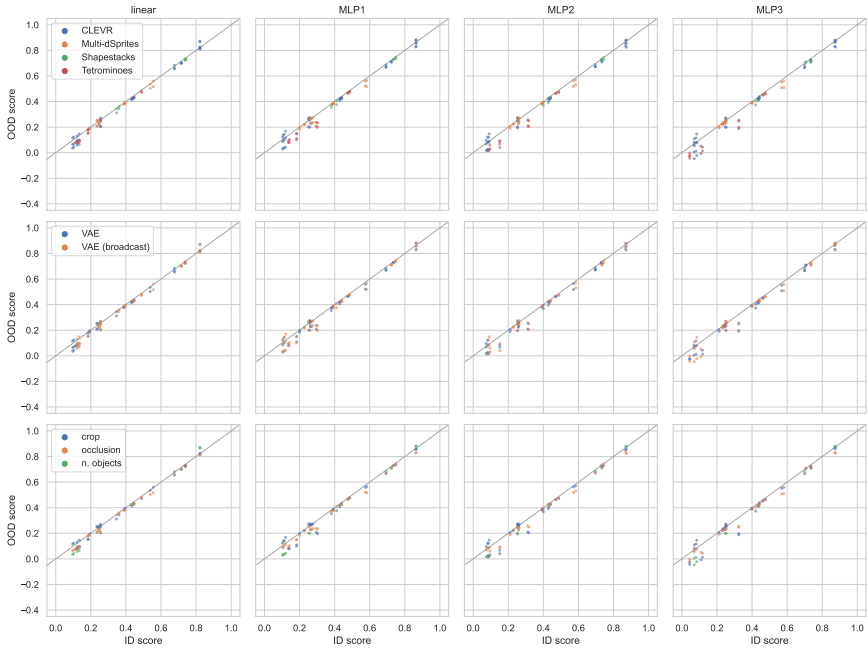


Figure C.25. Generalization of **distributed representations** in downstream prediction, using **deterministic matching** and **retraining** the downstream model after the distribution shift. Here the distribution shift affects **global properties** of the scene. On the x-axis: prediction performance (accuracy or R^2) for one object property on one dataset, averaged over all objects, on the original training set of the unsupervised object discovery model. On the y-axis: the same metric in OOD scenarios. Each data point corresponds to one representation model (e.g., MONet), one dataset, one object property, and one type of distribution shift. For each x (performance on one object feature in the training distribution, averaged over objects in a scene and over random seeds of the object-centric models) there are multiple y's, corresponding to different distribution shifts. In each row, we color-code the data according to dataset, model, or type of distribution shift. Each column shows analogous results for each of the 4 considered downstream models for property prediction (linear, and MLPs with up to 3 hidden layers).

C.4.3 Qualitative results

In Figs. C.26 to C.30 we show the reconstruction and segmentation performance of a selection of object-centric models on a random subset of held-out test images, for all 5 datasets. We select one object-centric model per type (MONet, Slot Attention, GENESIS, and SPACE) based on the ARI score on the validation set. The images we show were not used for model selection. For each model we show the following:

- *Input and reconstructed images.*
- *Ground-truth and inferred segmentation maps.* Here we use a set of 8 colors and assign each object (or slot) to a color. If there are more than 8 slots, we loop over the 8 colors again (this does not happen here, except in SPACE, where it is not an issue in practice). Rather than taking hard masks, we treat the masks as "soft", such that a pixel's color is a weighted mean of the 8 colors according to the masks. This is evident in Slot Attention, which typically splits the background smoothly across slots (consistently with the qualitative results shown in Locatello et al. (2020d)). For clarity, we match (with the Hungarian algorithm) the colors of the ground-truth and predicted masks using the cosine distance (1 minus the cosine similarity) between masks.
- *Slot-wise reconstructions.* Each column corresponds to a slot in the object-centric representation of the model. Here we show the entire slot reconstruction with the inferred slot mask as alpha (transparency) channel. The overall reconstruction is the sum of these images. Since SPACE has in total up to 69 slots in our experiments ($K = 5$ background slots, and a grid of foreground slots of size $G \times G$ with $G = 8$), it is impractical to show all slots here. We choose instead to show the 10 most salient slots, selected according to the average mask value over the image. This number is sufficient as most slots are unused. When selecting slots this way, the selected slots are shown in their original order (in SPACE, the background slots are appended to the foreground slots).

For completeness, in Fig. C.31 we show inputs and reconstructions for one VAE baseline per type (convolutional and broadcast decoder), selected using the reconstruction MSE on the validation set.

Finally, Figs. C.32 to C.36 show input–reconstruction pairs for each dataset, model type, and distribution shift. Note that the comparison is not necessarily fair, since

object-centric models were chosen using the validation ARI on the training distribution, while VAEs were chosen in a similar way but using the MSE. However, these qualitative results can still be highly informative. We report some examples:

- Most object-centric models are relatively robust to shifts affecting a single object, as discussed in the main text based on quantitative results.

- On the other hand, they are often not robust to global shifts, especially when cropping and enlarging the scene.

- MONet achieves relatively good reconstructions even out of distribution, probably because images are segmented mostly based on color. This was suggested by Papa, Winther, and Dittadi (2022), where the models are trained on objects with style transfer. However, we conjecture the behavior may be the same in our case, and that the argument should also apply to other distribution shifts, as seen by the relatively accurate reconstructions under both single-object and global distribution shifts. Note that, while reconstructions are potentially more accurate than for other models, this does not mean that MONet has segmented the object correctly.

- Although its ARI score does not decrease significantly, Slot Attention may not always handle more objects than in the training distribution, even when the number of slots in the model is increased. This is consistent with the results reported by Locatello et al. (2020d), and increasing the number of Slot Attention iterations at test time seems to be a promising approach (Locatello et al., 2020d, Fig. 2).

- VAEs seem to be relatively good at generalizing to a greater number of objects in CLEVR. In particular, they reconstruct images with the correct number of objects, although a few details of the objects may not be inferred correctly (e.g. an object may be reconstructed with the wrong size, color, or shape). This is surprising, since VAEs do not have any inductive bias for this, and the fact that the encoder is OOD (i.e., the encoder input is OOD w.r.t. the distribution used to train the encoder itself) might lead us to expect poor generalization capabilities, as discussed by Dittadi et al. (2021b) and Dittadi et al. (2022b) in the "OOD2" case. On the other hand, some object-centric models are remarkably robust to this shift (in particular SPACE, as confirmed by the ARI in Fig. 7.8).

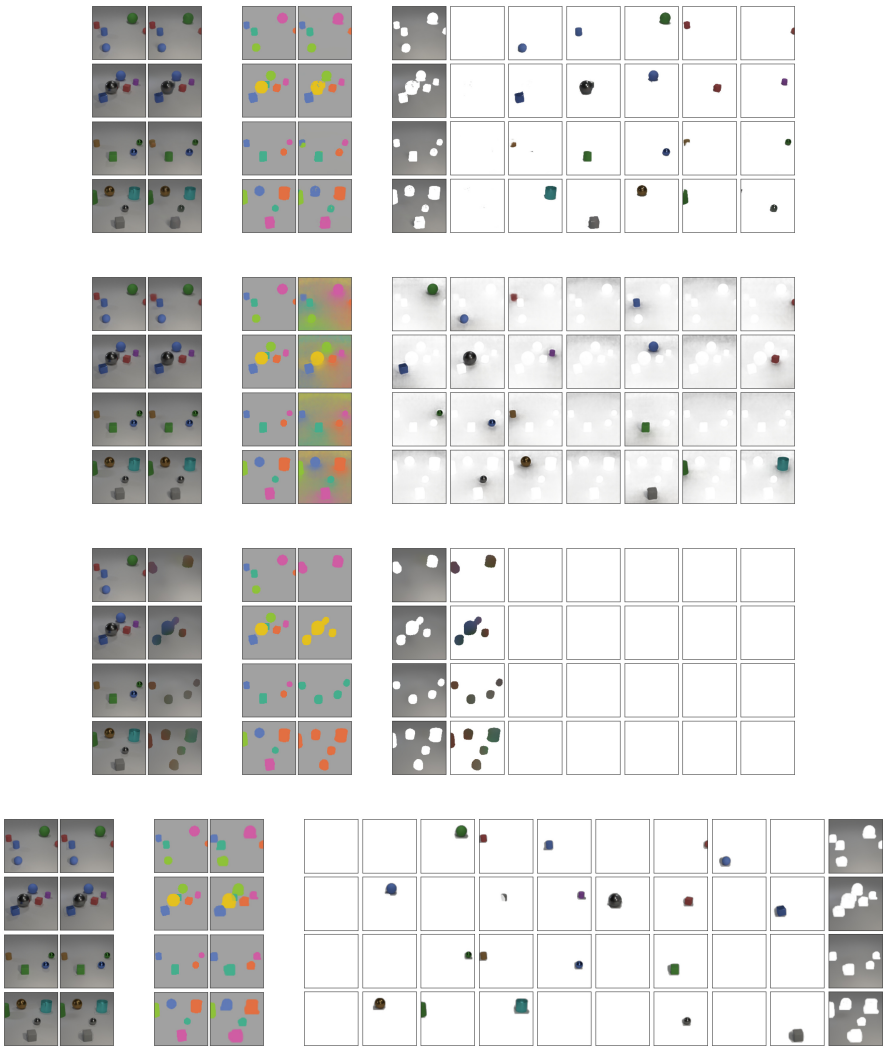


Figure C.26. Reconstruction and segmentation of 4 random images from the held-out test set of **CLEVR6**. Top to bottom: MONet, Slot Attention, GENESIS, SPACE. Left to right: input, reconstruction, ground-truth masks, predicted (soft) masks, slot-wise reconstructions (masked with the predicted masks). As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. For each model type, we visualize the specific model with the highest ARI score in the *validation* set. The images shown here are from the *test* set and were not used for model selection.

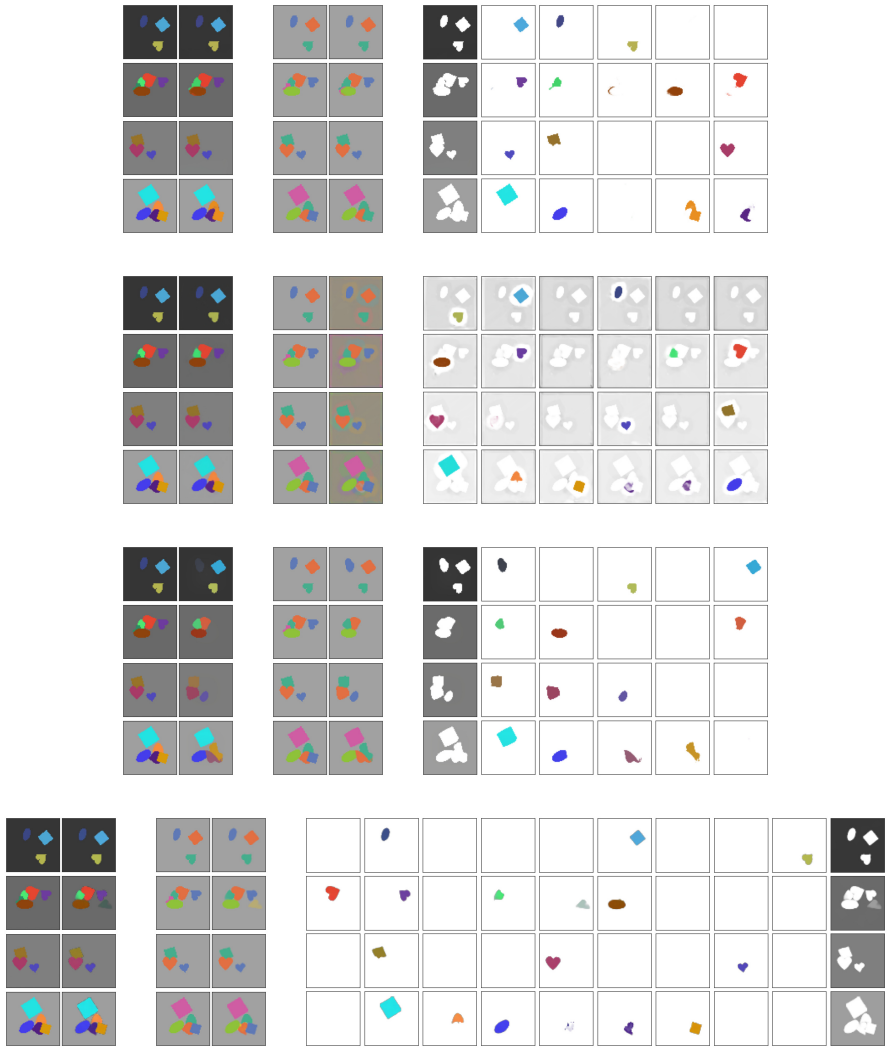


Figure C.27. Reconstruction and segmentation of 4 random images from the held-out test set of **Multi-dSprites**. Top to bottom: MONet, Slot Attention, GENESIS, SPACE. Left to right: input, reconstruction, ground-truth masks, predicted (soft) masks, slot-wise reconstructions (masked with the predicted masks). As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. For each model type, we visualize the specific model with the highest ARI score in the *validation* set. The images shown here are from the *test* set and were not used for model selection.

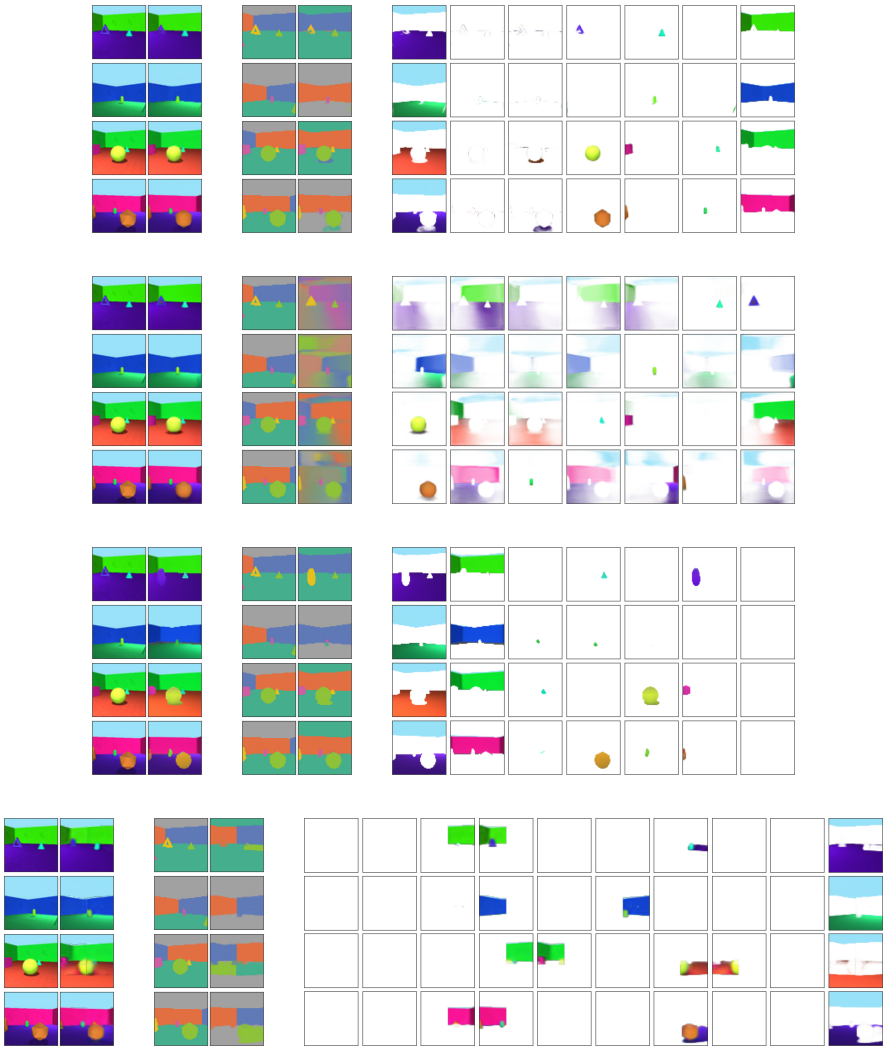


Figure C.28. Reconstruction and segmentation of 4 random images from the held-out test set of **Objects Room**. Top to bottom: MONet, Slot Attention, GENESIS, SPACE. Left to right: input, reconstruction, ground-truth masks, predicted (soft) masks, slot-wise reconstructions (masked with the predicted masks). As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. For each model type, we visualize the specific model with the highest ARI score in the *validation* set. The images shown here are from the *test* set and were not used for model selection.

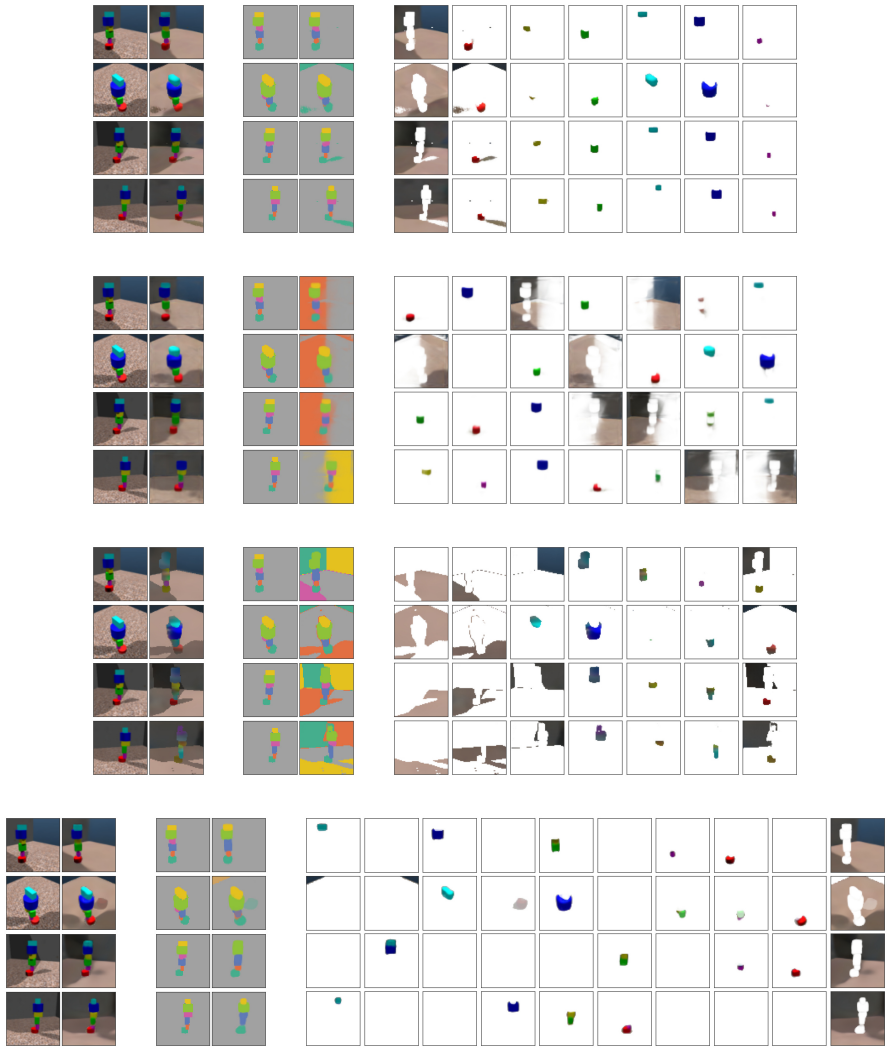


Figure C.29. Reconstruction and segmentation of 4 random images from the held-out test set of **Shapestacks**. Top to bottom: MONet, Slot Attention, GENESIS, SPACE. Left to right: input, reconstruction, ground-truth masks, predicted (soft) masks, slot-wise reconstructions (masked with the predicted masks). As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. For each model type, we visualize the specific model with the highest ARI score in the *validation* set. The images shown here are from the *test* set and were not used for model selection.

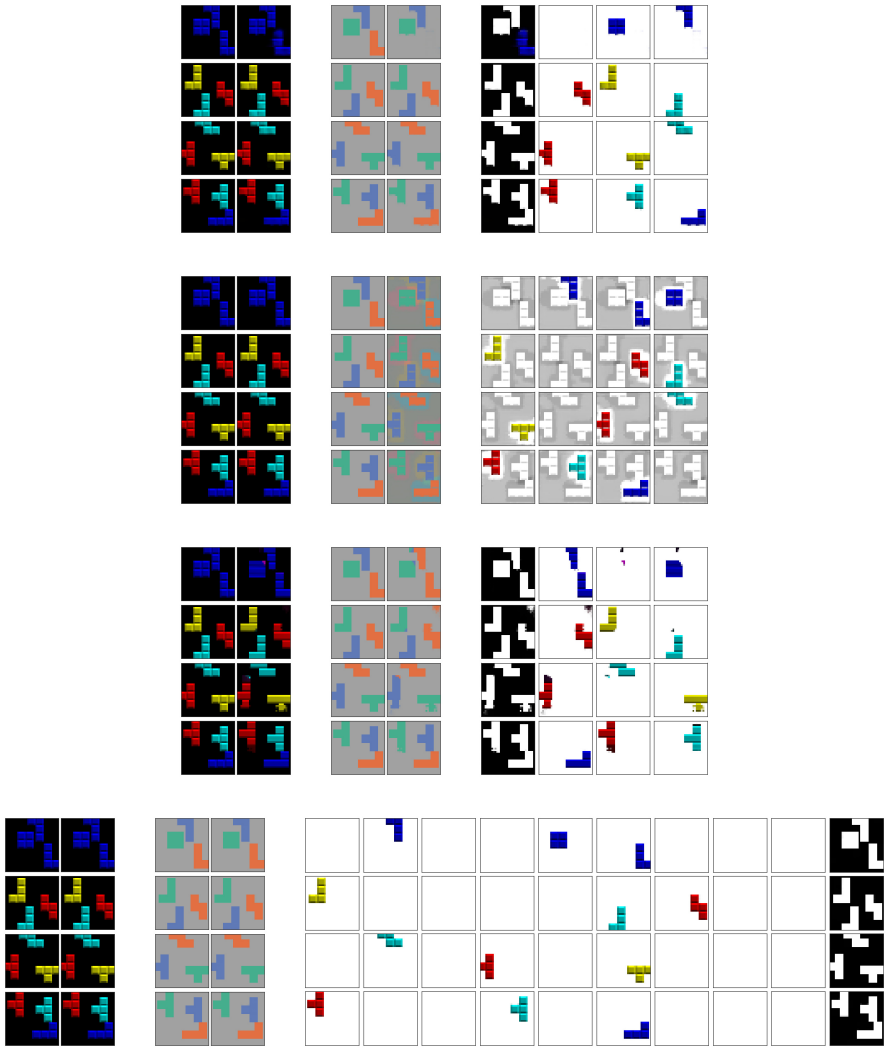


Figure C.30. Reconstruction and segmentation of 4 random images from the held-out test set of **Tetrominoes**. Top to bottom: MONet, Slot Attention, GENESIS, SPACE. Left to right: input, reconstruction, ground-truth masks, predicted (soft) masks, slot-wise reconstructions (masked with the predicted masks). As explained in the text, for SPACE we select the 10 most salient slots using the predicted masks. For each model type, we visualize the specific model with the highest ARI score in the *validation* set. The images shown here are from the *test* set and were not used for model selection.

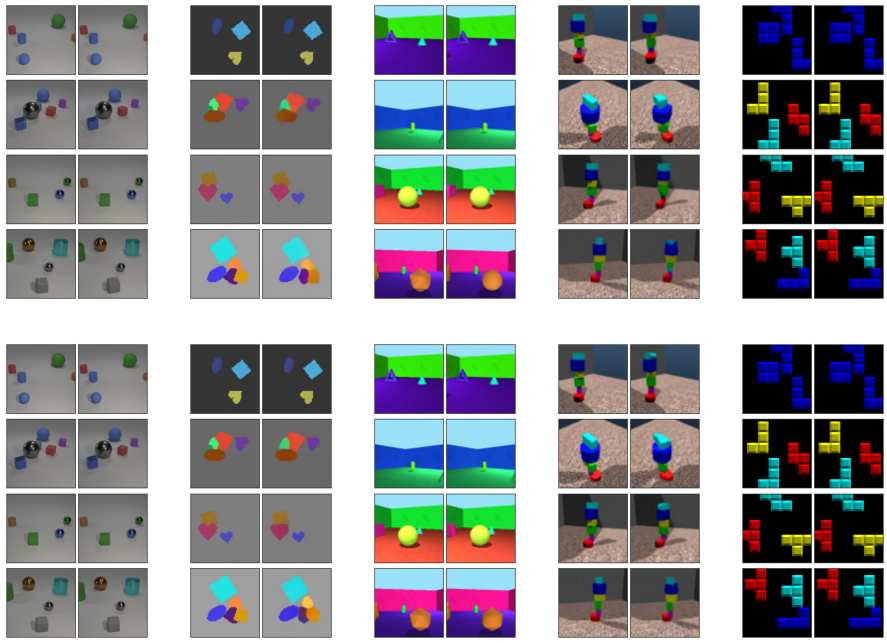


Figure C.31. Input–reconstruction pairs of 4 random images from the held-out *test* set of all 5 datasets, for the VAE model with convolutional (top) and broadcast (bottom) decoder. Each VAE type was trained with 5 random seeds, and for each type we show here the model with the lowest MSE on the *validation* set. The images shown here are from the *test* set and were not used for model selection. For each image, we show the input on the left and the reconstruction on the right. As these are not slot-based models, segmentation masks and slot-wise reconstructions are not available.

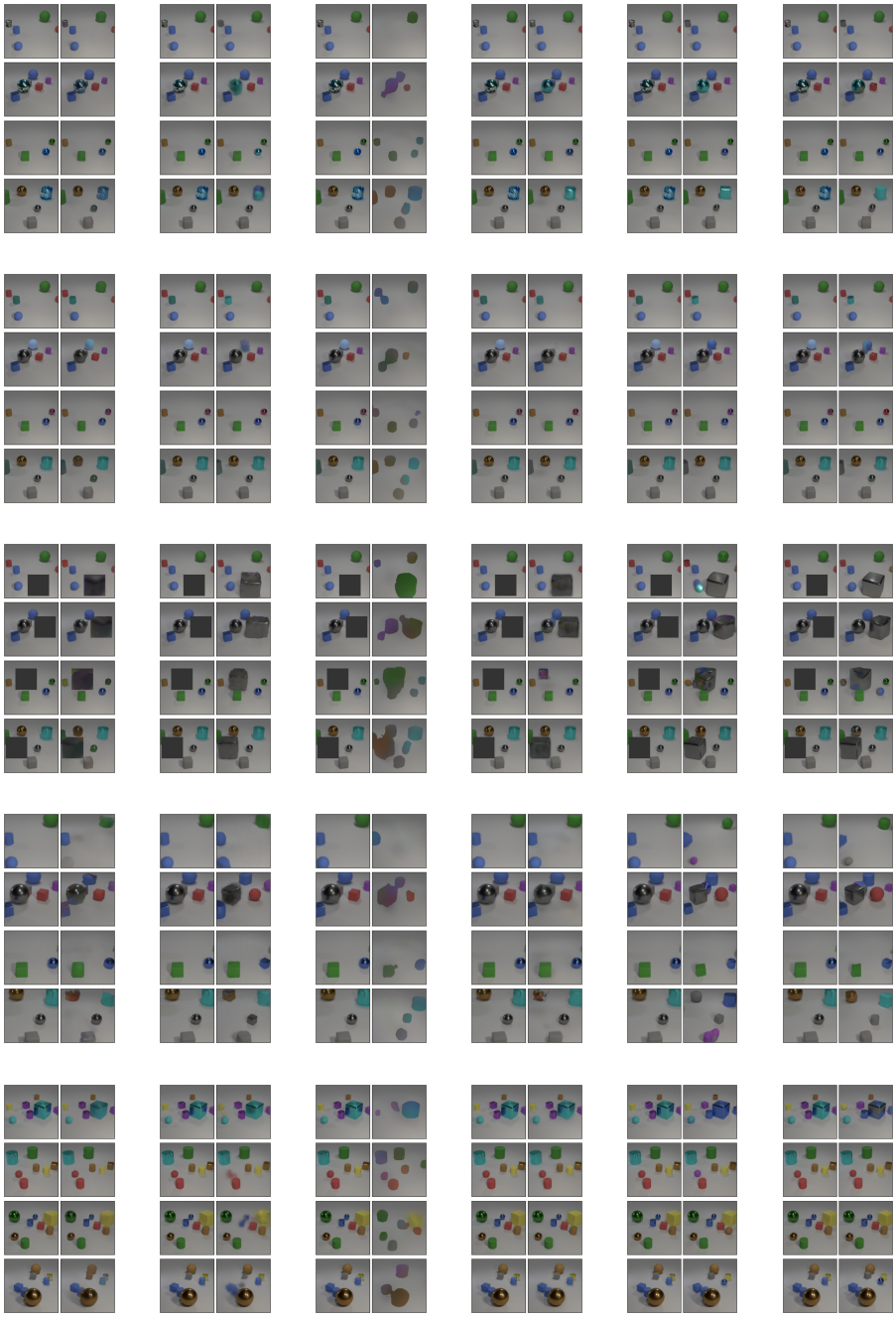


Figure C.32. Inputs and reconstructions for OOD images in CLEVR. Columns from left to right: MONet, Slot Attention, GENESIS, SPACE, convolutional decoder VAE, broadcast decoder VAE. Rows from top to bottom: object style, object color, occlusion, crop, number of objects.

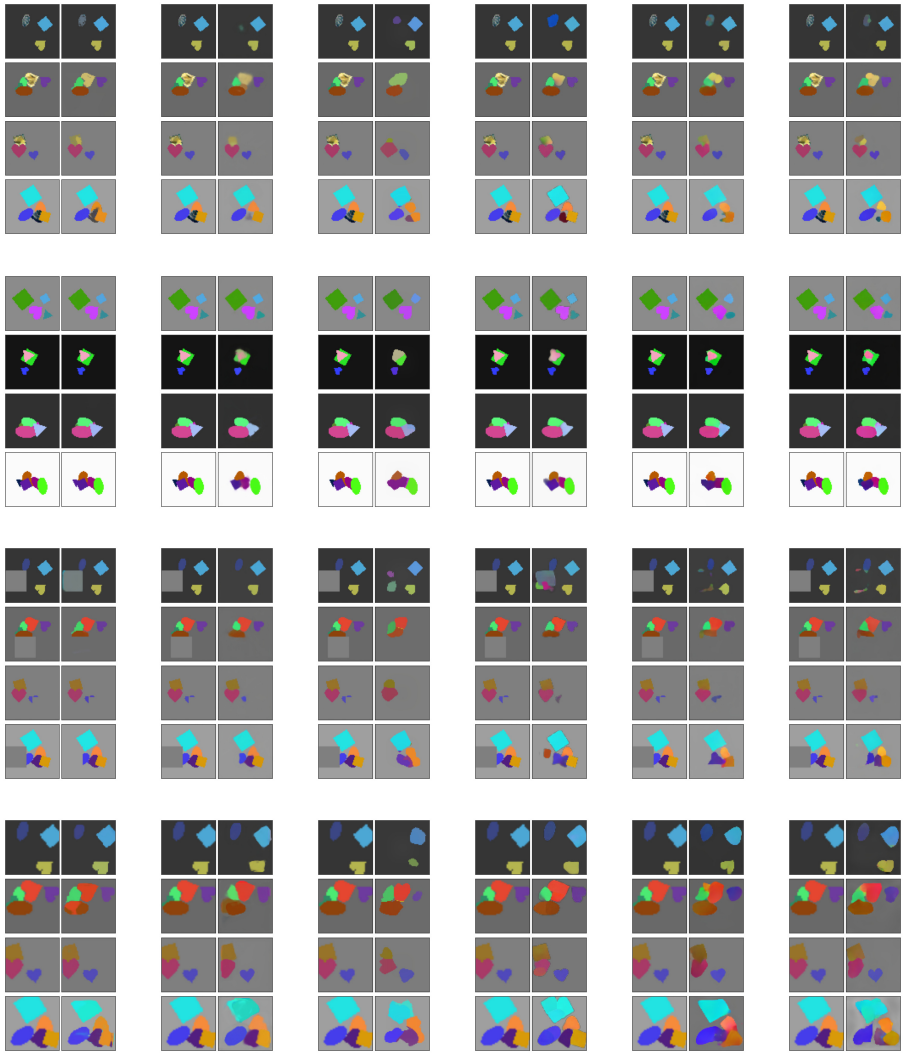


Figure C.33. Inputs and reconstructions for OOD images in Multi-dSprites. Columns from left to right: MONet, Slot Attention, GENESIS, SPACE, convolutional decoder VAE, broadcast decoder VAE. Rows from top to bottom: object style, object shape, occlusion, crop.

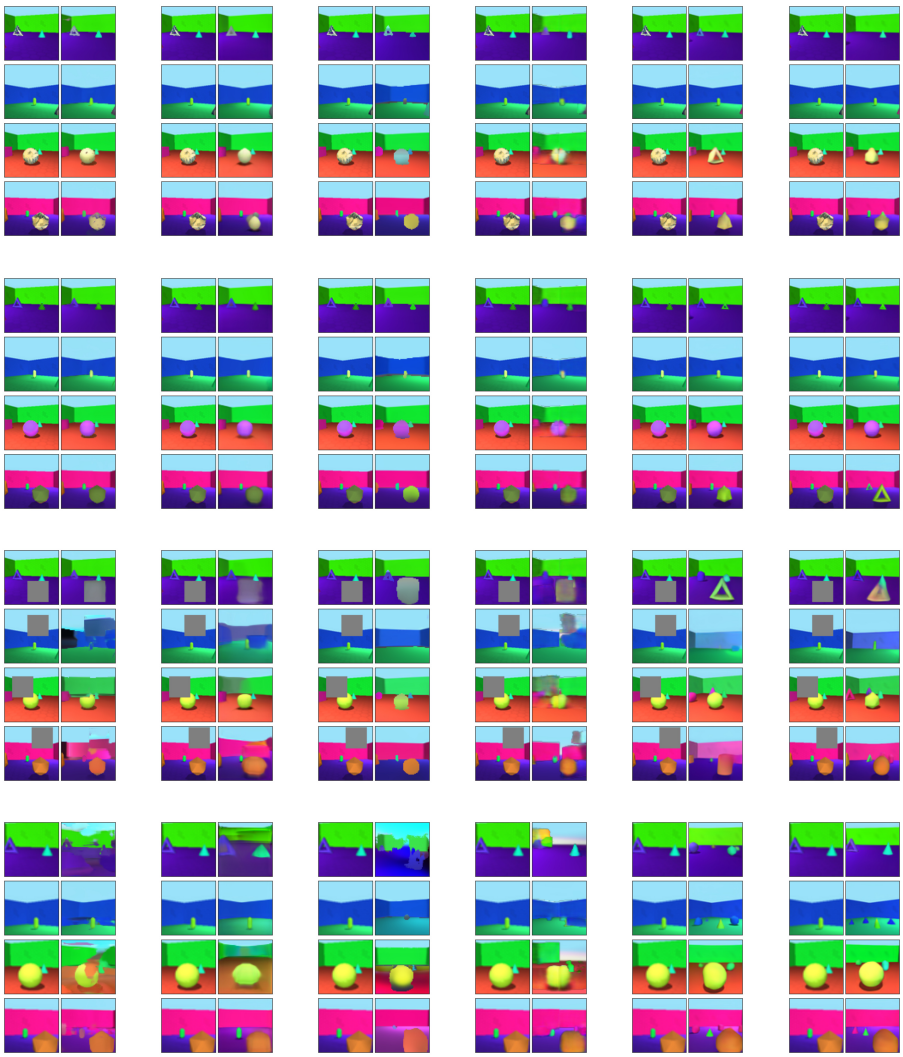


Figure C.34. Inputs and reconstructions for OOD images in Objects Room. Columns from left to right: MONet, Slot Attention, GENESIS, SPACE, convolutional decoder VAE, broadcast decoder VAE. Rows from top to bottom: object style, object color, occlusion, crop.

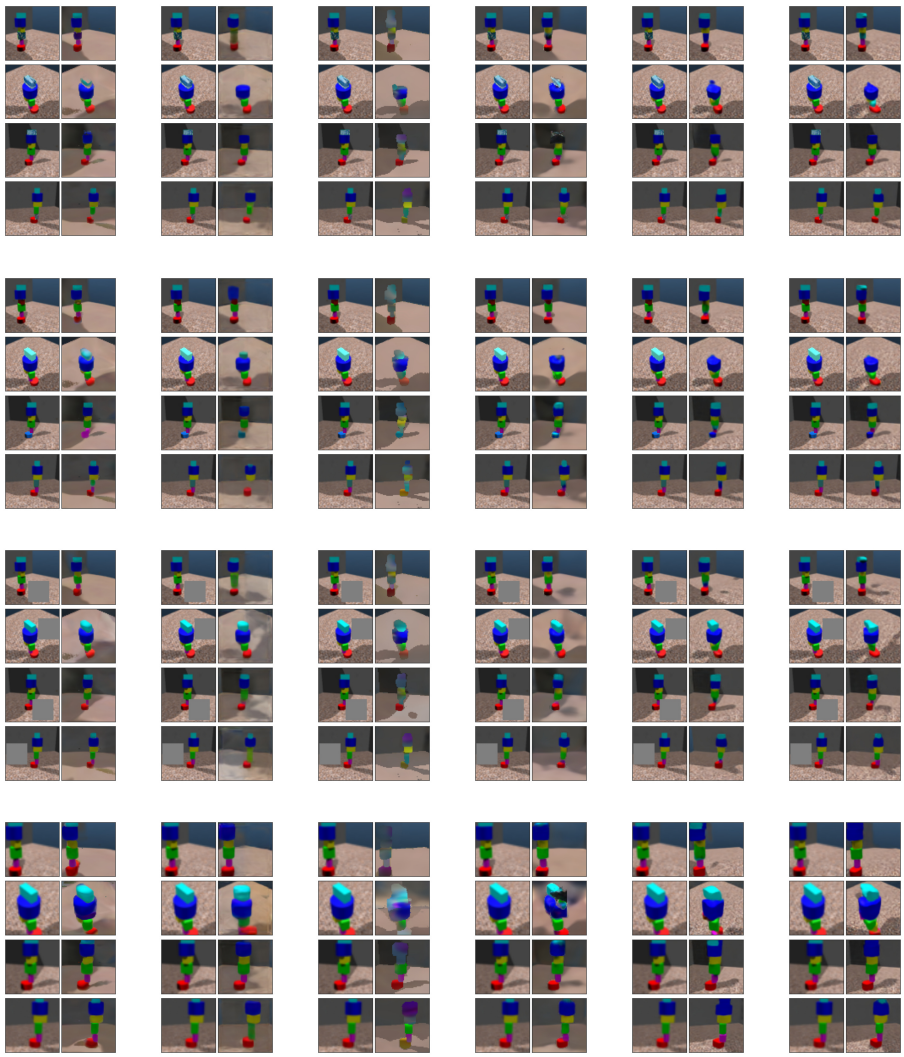


Figure C.35. Inputs and reconstructions for OOD images in Shapestacks. Columns from left to right: MONet, Slot Attention, GENESIS, SPACE, convolutional decoder VAE, broadcast decoder VAE. Rows from top to bottom: object style, object color, occlusion, crop.

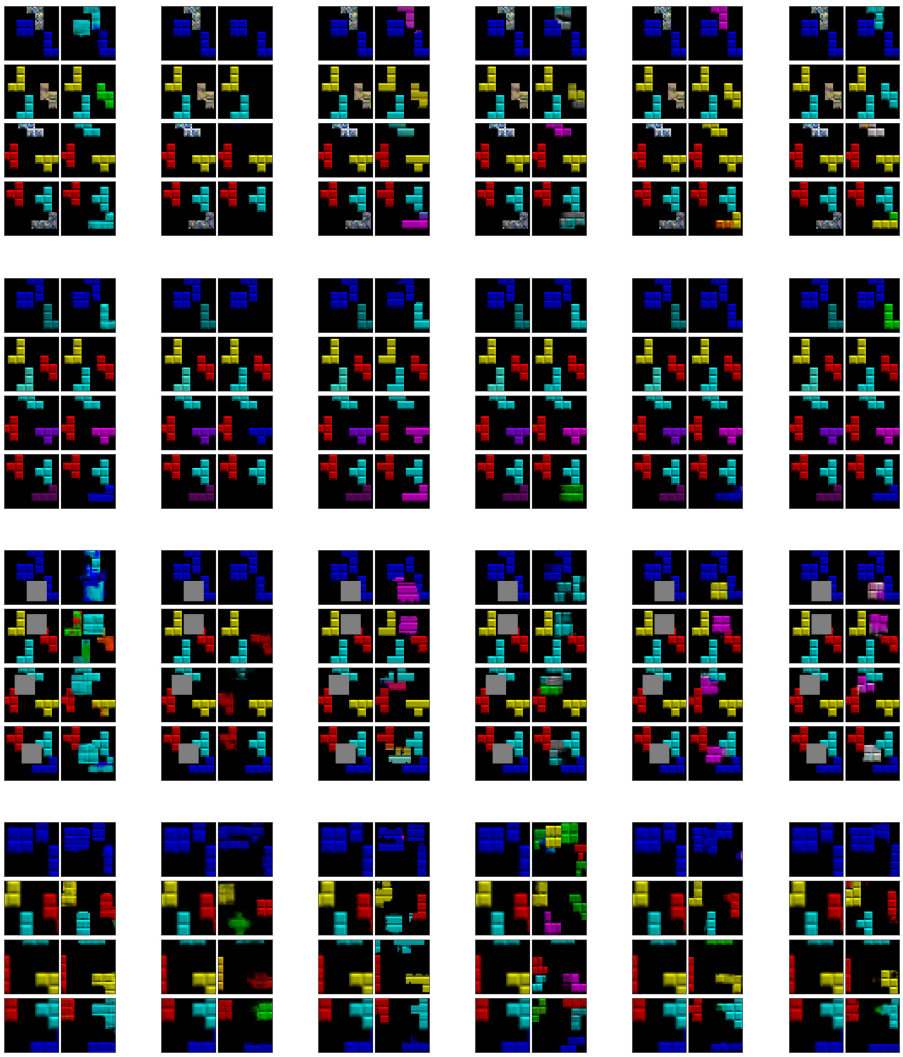


Figure C.36. Inputs and reconstructions for OOD images in Tetrominoes. Columns from left to right: MONet, Slot Attention, GENESIS, SPACE, convolutional decoder VAE, broadcast decoder VAE. Rows from top to bottom: object style, object color, occlusion, crop.

Bibliography

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. (2016). "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." In: *arXiv preprint arXiv:1603.04467* (cit. on p. 55).

Abid, Abubakar, Maheen Farooqi, and James Zou (2021). "Persistent anti-muslim bias in large language models." In: *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 298–306 (cit. on p. 13).

Achille, Alessandro and Stefano Soatto (2018). "Emergence of Invariance and Disentanglement in Deep Representations." In: *Journal of Machine Learning Research* 19.50, pp. 1–34 (cit. on pp. 6, 10, 20).

Adel, Tameem, Zoubin Ghahramani, and Adrian Weller (2018). "Discovering interpretable representations for both deep generative and discriminative models." In: *International Conference on Machine Learning*, pp. 50–59 (cit. on pp. 11, 64).

Aha, David W (1991). "Incremental constructive induction: An instance-based approach." In: *Machine Learning Proceedings 1991*. Elsevier, pp. 117–121 (cit. on p. 6).

Ahmed, Amr, Kai Yu, Wei Xu, Yihong Gong, and Eric Xing (2008). "Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks." In: *European Conference on Computer Vision*. Springer, pp. 69–82 (cit. on p. 8).

Ahmed, Ossama, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Manuel Wüthrich, Yoshua Bengio, Bernhard Schölkopf, and Stefan Bauer (2021). "Causalworld: A robotic manipulation benchmark for causal structure and transfer learning." In: *In-*

ternational Conference for Learning Representations (cit. on pp. 9, 30, 38, 48, 80, 81, 84, 86, 87, 95, 135, 136).

Akkaya, Ilge, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. (2019). "Solving rubik's cube with a robot hand." In: *arXiv preprint arXiv:1910.07113* (cit. on p. 95).

Alemi, Alexander, Ben Poole, Ian Fischer, Joshua Dillon, Rif A Saurous, and Kevin Murphy (2018). "Fixing a broken ELBO." In: *International Conference on Machine Learning*. PMLR, pp. 159–168 (cit. on pp. 18, 19).

Anand, Ankesh, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm (2019). "Unsupervised state representation learning in atari." In: *arXiv preprint arXiv:1906.08226* (cit. on p. 80).

Andrychowicz, OpenAI: Marcin, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. (2020). "Learning dexterous in-hand manipulation." In: *The International Journal of Robotics Research* 39.1, pp. 3–20 (cit. on p. 67).

Anselmi, Fabio, Joel Z. Leibo, Lorenzo Rosasco, Jim Mutch, Andrea Tacchetti, and Tomaso A. Poggio (2016). "Unsupervised learning of invariant representations." In: *Theor. Comput. Sci.* 633, pp. 112–121 (cit. on p. 12).

Arbelaez, Pablo, Michael Maire, Charless Fowlkes, and Jitendra Malik (2010). "Contour detection and hierarchical image segmentation." In: *IEEE transactions on pattern analysis and machine intelligence* 33.5, pp. 898–916 (cit. on pp. 36, 103, 159).

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz (2019). "Invariant risk minimization." In: *arXiv preprint arXiv:1907.02893* (cit. on p. 67).

Asadi, Kavosh, David Abel, and Michael L Littman (2020). "Learning state abstractions for transfer in continuous control." In: *arXiv preprint arXiv:2002.05518* (cit. on p. 95).

Asai, Masataro and Alex Fukunaga (2018). "Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary." In: *Proceedings of the aaai conference on artificial intelligence*. Vol. 32. 1 (cit. on p. 30).

Ayton, Benjamin and Masataro Asai (2021). "Is Policy Learning Overrated?: Width-Based Planning and Active Learning for Atari." In: *arXiv preprint arXiv:2109.15310* (cit. on p. 30).

Azulay, Aharon and Yair Weiss (2019). "Why do deep convolutional networks generalize so poorly to small image transformations?" In: *Journal of Machine Learning Research* 20.184, pp. 1–25 (cit. on p. 80).

Baader, Franz and Werner Nutt (2003). "Basic description logics." In: *The description logic handbook: theory, implementation, and applications*, pp. 43–95 (cit. on p. 29).

Bach, Francis and Michael Jordan (2002). "Kernel independent component analysis." In: *Journal of Machine Learning Research* 3.7, pp. 1–48 (cit. on p. 66).

Bachlechner, Thomas, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W Cottrell, and Julian McAuley (2020). "Rezero is all you need: Fast convergence at large depth." In: *arXiv preprint arXiv:2003.04887* (cit. on pp. 69, 124, 126).

Baillargeon, Renee, Elizabeth S Spelke, and Stanley Wasserman (1985). "Object permanence in five-month-old infants." In: *Cognition* 20.3, pp. 191–208 (cit. on pp. 30, 98).

Ballard, Dana H. (1987). "Modular Learning in Neural Networks." In: *Proceedings of the 6th National Conference on Artificial Intelligence. Seattle, WA, USA, July 1987*. Ed. by Kenneth D. Forbus and Howard E. Shrobe. Morgan Kaufmann, pp. 279–284 (cit. on p. 6).

Barbu, Andrei, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz (2019). "ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models." In: *Advances in Neural Information Processing Systems*, pp. 9448–9458 (cit. on p. 80).

Barlow, Horace B (1989). "Unsupervised learning." In: *Neural computation* 1.3, pp. 295–311 (cit. on p. 18).

Barreto, André, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver (2017). "Successor features for transfer in reinforcement learning." In: *Advances in neural information processing systems*, pp. 4055–4065 (cit. on p. 80).

Battaglia, Peter, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and koray kavukcuoglu koray (2016). "Interaction Networks for Learning about Objects, Relations and Physics." In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett. Vol. 29. Curran Associates, Inc. (cit. on pp. 30, 98).

Battaglia, Peter W, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu (2018). "Relational inductive biases, deep learning, and graph networks." In: *arXiv preprint arXiv:1806.01261* (cit. on pp. 29, 30).

Belongie, Serge, Jitendra Malik, and Jan Puzicha (2001). "Matching shapes." In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001.* Vol. 1. IEEE, pp. 454–461 (cit. on p. 6).

Bender, Emily M, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell (2021). "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? ." In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 610–623 (cit. on p. 13).

Bengio, Yoshua (2009). "Learning Deep Architectures for AI." In: *Found. Trends Mach. Learn.* 2.1, pp. 1–127. DOI: 10.1561/2200000006. URL: https://doi.org/10.1561/2200000006 (cit. on pp. 9, 10).

Bengio, Yoshua, Aaron Courville, and Pascal Vincent (2013). "Representation learning: A review and new perspectives." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1798–1828 (cit. on pp. 5–11, 13, 18, 19, 80, 98).

Bengio, Yoshua, Pascal Lamblin, Dan Popovici, and Hugo Larochelle (2006). "Greedy layer-wise training of deep networks." In: *Advances in neural information processing systems* 19 (cit. on p. 8).

Berner, Christopher, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang (2019). "Dota 2 with Large Scale Deep Reinforcement Learning." In: *arXiv:1912.06680* (cit. on pp. 30, 98).

Bing, Simon, Andrea Dittadi, Stefan Bauer, and Patrick Schwab (2022). "Conditional Generation of Medical Time Series for Extrapolation to Underrepresented Populations." In: *arXiv preprint arXiv:2201.08186* (cit. on p. xiv).

Bishop, Chris M (1995). "Training with noise is equivalent to Tikhonov regularization." In: *Neural computation* 7.1, pp. 108–116 (cit. on pp. 69, 76).

Bommasani, Rishi, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. (2021). "On the opportunities and risks of foundation models." In: *arXiv preprint arXiv:2108.07258* (cit. on p. 9).

Bordes, Antoine, Sumit Chopra, and Jason Weston (2014). "Question answering with subgraph embeddings." In: *arXiv preprint arXiv:1406.3676* (cit. on p. 7).

Bouchacourt, Diane, Ryota Tomioka, and Sebastian Nowozin (2018). "Multi-level variational autoencoder: Learning disentangled representations from grouped observa-

tions." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1 (cit. on pp. 26, 27).

Bowers, Jeffrey S, Ivan I Vankov, Markus F Damian, and Colin J Davis (2014). "Neural networks learn highly selective representations in order to overcome the superposition catastrophe." In: *Psychological review* 121.2, p. 248 (cit. on pp. 31, 99).

Bowman, Samuel R, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio (2015). "Generating sentences from a continuous space." In: *arXiv preprint arXiv:1511.06349* (cit. on p. 69).

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020a). "Language models are few-shot learners." In: *Advances in neural information processing systems* 33, pp. 1877–1901 (cit. on pp. 7, 13).

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei (2020b). "Language Models are Few-Shot Learners." In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (cit. on p. 9).

Buchanan, B, G Sutherland, and EA Feigenbaum (1969). "Heuristic DENDRAL: A program for generating explanatory hypotheses." In: *Organic Chemistry* (cit. on p. 29).

Burda, Yuri, Roger Grosse, and Ruslan Salakhutdinov (2015). "Importance weighted autoencoders." In: *arXiv preprint arXiv:1509.00519* (cit. on p. 17).

Burgess, Christopher P, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner (2018). "Understanding disentangling in beta-VAE." In: *arXiv preprint arXiv:1804.03599* (cit. on pp. 19, 20, 66, 82).

Burgess, Christopher P, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner (2019). "Monet: Unsupervised scene decomposition and representation." In: *arXiv preprint arXiv:1901.11390* (cit. on pp. 33, 34, 98, 102, 113, 143, 145, 152, 157).

Carlini, Nicholas, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et

al. (2021). “Extracting training data from large language models.” In: *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650 (cit. on p. 13).

Caruana, Rich (1997). “Multitask learning.” In: *Machine learning* 28.1, pp. 41–75 (cit. on p. 10).

Chaabouni, Rahma, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni (2020). “Compositionality and Generalization in Emergent Languages.” In: *ACL 2020-8th annual meeting of the Association for Computational Linguistics* (cit. on p. 95).

Chapelle, Olivier, Bernhard Schölkopf, and Alexander Zien, eds. (2006). *Semi-Supervised Learning*. The MIT Press. ISBN: 9780262033589. DOI: 10.7551/mitpress/9780262033589.001.0001. URL: https://doi.org/10.7551/mitpress/9780262033589.001.0001 (cit. on p. 9).

Chartsias, Agisilaos, Thomas Joyce, Giorgos Papanastasiou, Scott Semple, Michelle Williams, David Newby, Rohan Dharmakumar, and Sotirios A Tsaftaris (2018). “Factorised spatial representation learning: Application in semi-supervised myocardial segmentation.” In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 490–498 (cit. on p. 77).

Chen, Chang, Fei Deng, and Sungjin Ahn (2020). “Learning to infer 3d object models from images.” In: *arXiv preprint arXiv:2006.06130* (cit. on p. 98).

Chen, Mark, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever (2020a). “Generative pretraining from pixels.” In: *International Conference on Machine Learning*. PMLR, pp. 1691–1703 (cit. on p. 8).

Chen, Mickaël, Thierry Artières, and Ludovic Denoyer (2019a). “Unsupervised Object Segmentation by Redrawing.” In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. (cit. on p. 113).

– (2019b). “Unsupervised object segmentation by redrawing.” In: *Advances in neural information processing systems* 32 (cit. on p. 32).

Chen, Tian Qi, Xuechen Li, Roger Grosse, and David Duvenaud (2018). “Isolating Sources of Disentanglement in Variational Autoencoders.” In: *Advances in Neural Information Processing Systems* (cit. on pp. 11, 21, 24, 66, 86, 92, 100).

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (July 2020b). “A Simple Framework for Contrastive Learning of Visual Representations.” In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 1597–1607 (cit. on pp. 8, 9).

Chen, Xi, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel (2016). "Infogan: Interpretable representation learning by information maximizing generative adversarial nets." In: *Advances in Neural Information Processing Systems* (cit. on p. 19).

Chen, Yukun, Andrea Dittadi, Frederik Träuble, Stefan Bauer, and Bernhard Schölkopf (2021). "Boxhead: A Dataset for Learning Hierarchical Representations." In: *SVRHM 2021 Workshop @ NeurIPS* (cit. on pp. xiv, 19, 84).

Chira, Darius, Ilian Haralampiev, Ole Winther, Andrea Dittadi, and Valentin Liévin (2022). "Image Super-Resolution With Deep Variational Autoencoders." In: *arXiv preprint arXiv:2203.09445* (cit. on p. xiv).

Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. (2022). "PaLM: Scaling Language Modeling with Pathways." In: *arXiv preprint arXiv:2204.02311* (cit. on p. 13).

Ciresan, Dan Claudiu, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber (2011). "Flexible, high performance convolutional neural networks for image classification." In: *Twenty-second international joint conference on artificial intelligence* (cit. on p. 7).

Cobbe, Karl, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman (2019). "Quantifying generalization in reinforcement learning." In: *International Conference on Machine Learning*. PMLR, pp. 1282–1289 (cit. on pp. 80, 95).

Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa (2011). "Natural language processing (almost) from scratch." In: *Journal of machine learning research* 12.ARTICLE, pp. 2493–2537 (cit. on p. 7).

Comon, Pierre (1994). "Independent component analysis, a new concept?" In: *Signal Processing* 36.3, pp. 287–314 (cit. on p. 66).

Cowan, Nelson (2001). "The magical number 4 in short-term memory: A reconsideration of mental storage capacity." In: *Behavioral and Brain Sciences* 24.1, pp. 87–114 (cit. on p. 29).

Crawford, Eric and Joelle Pineau (2019). "Spatially invariant unsupervised object detection with convolutional neural networks." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 3412–3420 (cit. on pp. 98, 114).

– (2020). "Exploiting spatial invariance for scalable unsupervised object tracking." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34, pp. 3684–3692 (cit. on p. 114).

Cresswell, Stephen N, Thomas L McCluskey, and Margaret M West (2013). "Acquiring planning domain models using LOCM." In: *The Knowledge Engineering Review* 28.2, pp. 195–213 (cit. on p. 29).

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection." In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee, pp. 886–893 (cit. on p. 6).

Dayan, Peter (1993). "Improving generalization for temporal difference learning: The successor representation." In: *Neural Computation* 5.4, pp. 613–624 (cit. on p. 67).

Dehaene, Stanislas (2020). *How We Learn: Why Brains Learn Better than Any Machine ... for Now*. New York: Viking (cit. on pp. 30, 98).

Deng, Fei, Zhuo Zhi, Donghun Lee, and Sungjin Ahn (2021). "Generative Scene Graph Networks." In: *International Conference on Learning Representations* (cit. on p. 114).

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "ImageNet: A large-scale hierarchical image database." In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 248–255 (cit. on p. 8).

Denton, Emily L, Soumith Chintala, Arthur Szlam, and Rob Fergus (2015). "Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks." In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. (cit. on p. 7).

Devereux, Barry J, Lorraine K Tyler, Jeroen Geertzen, and Billi Randall (2014). "The Centre for Speech, Language and the Brain (CSLB) concept property norms." In: *Behavior research methods* 46.4, pp. 1119–1127 (cit. on p. 11).

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, pp. 4171–4186 (cit. on p. 9).

Dittadi, Andrea, Thomas Bolander, and Ole Winther (2018). "Learning to Plan from Raw Data in Grid-based Games." In: *GCAI-2018. 4th Global Conference on Artificial Intelligence*. Ed. by Daniel Lee, Alexander Steen, and Toby Walsh. Vol. 55. EPiC Series in Computing. EasyChair, pp. 54–67. DOI: 10.29007/s8jk (cit. on pp. xv, 29).

Dittadi, Andrea, Frederik K Drachmann, and Thomas Bolander (2021). "Planning from Pixels in Atari with Learned Symbolic Representations." In: *Proceedings of*

the AAAI Conference on Artificial Intelligence. Vol. 35. 6, pp. 4941–4949 (cit. on pp. xv, 9, 30, 80).

Dittadi, Andrea, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Thomas J Cashman, and Jamie Shotton (2021a). "Full-Body Motion From a Single Head-Mounted Device: Generating SMPL Poses From Partial Observations." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11687–11697 (cit. on p. xiv).

Dittadi, Andrea, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, and Francesco Locatello (2022a). "Generalization and Robustness Implications in Object-Centric Learning." In: *International Conference on Machine Learning*. PMLR (cit. on pp. xiii, 2, 9, 51, 53, 95).

Dittadi, Andrea, Frederik Träuble, Francesco Locatello, Manuel Wuthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, and Bernhard Schölkopf (2021b). "On the Transfer of Disentangled Representations in Realistic Settings." In: *International Conference on Learning Representations* (cit. on pp. xiii, 1, 9, 11, 12, 19, 28, 38, 81, 83–87, 89, 91–94, 113, 135–138, 193).

Dittadi, Andrea, Frederik Träuble, Manuel Wuthrich, Felix Widmaier, Peter Vincent Gehler, Ole Winther, Francesco Locatello, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer (2022b). "The Role of Pretrained Representations for the OOD Generalization of Reinforcement Learning Agents." In: *International Conference on Learning Representations* (cit. on pp. xiii, 1, 9, 12, 19, 28, 113, 193).

Dittadi, Andrea and Ole Winther (2019). "LAVAE: Disentangling Location and Appearance." In: *NeurIPS Workshop on Perception as Generative Reasoning* (cit. on pp. xv, 18, 31, 114).

Djolonga, Josip, Jessica Yung, Michael Tschannen, Rob Romijnders, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Matthias Minderer, Alexander D'Amour, Dan Moldovan, et al. (2021). "On robustness and transferability of convolutional neural networks." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16458–16468 (cit. on pp. 80, 93).

Doersch, Carl, Abhinav Gupta, and Alexei A Efros (2015). "Unsupervised visual representation learning by context prediction." In: *Proceedings of the IEEE international conference on computer vision*, pp. 1422–1430 (cit. on p. 8).

Donahue, Jeff and Karen Simonyan (2019). "Large scale adversarial representation learning." In: *Advances in Neural Information Processing Systems* 32 (cit. on p. 8).

Dosovitskiy, Alexey, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox (2014). "Discriminative unsupervised feature learning with convolutional neural networks." In: *Advances in neural information processing systems* 27 (cit. on p. 8).

Drew, McDermott (1998). *The planning domain definition language manual*. Tech. rep. 1165. Yale Computer Science, New Hawen, CT (cit. on p. 29).

Dulac-Arnold, Gabriel, Daniel Mankowitz, and Todd Hester (2019). "Challenges of real-world reinforcement learning." In: *arXiv preprint arXiv:1904.12901* (cit. on p. 95).

Eastwood, Cian and Christopher KI Williams (2018). "A framework for the quantitative evaluation of disentangled representations." In: *International Conference on Learning Representations* (cit. on pp. 11, 22–25, 66, 92, 100).

Eldan, Ronen and Ohad Shamir (June 2016). "The Power of Depth for Feedforward Neural Networks." In: *29th Annual Conference on Learning Theory*. Ed. by Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir. Vol. 49. Proceedings of Machine Learning Research. Columbia University, New York, New York, USA: PMLR, pp. 907–940 (cit. on p. 7).

Engelcke, Martin, Oiwi Parker Jones, and Ingmar Posner (2020). "Reconstruction Bottlenecks in Object-Centric Generative Models." In: *arXiv preprint arXiv:2007.06245* (cit. on p. 113).

Engelcke, Martin, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner (2020). "GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations." In: *ICLR 2020* (cit. on pp. 33, 34, 36, 98, 102, 103, 113, 144, 147, 157, 159).

Engelcke, Martin, Oiwi Parker Jones, and Ingmar Posner (2021). "GENESIS-V2: Inferring Unordered Object Representations without Iterative Refinement." In: *arXiv preprint arXiv:2104.09958* (cit. on p. 113).

Erhan, Dumitru, Aaron Courville, Yoshua Bengio, and Pascal Vincent (2010). "Why does unsupervised pre-training help deep learning?" In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 201–208 (cit. on p. 9).

Eslami, SM Ali, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Geoffrey E Hinton, et al. (2016). "Attend, infer, repeat: Fast scene understanding with generative models." In: *Advances in Neural Information Processing Systems*, pp. 3225–3233 (cit. on pp. 12, 98, 113).

Eslami, SM Ali, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. (2018). "Neural scene representation and rendering." In: *Science* 360.6394, pp. 1204–1210 (cit. on pp. 80, 81, 153).

Esmaeili, Babak, Hao Wu, Sarthak Jain, Alican Bozkurt, N Siddharth, Brooks Paige, Dana H Brooks, Jennifer Dy, and Jan-Willem Meent (2019a). "Structured dis-

entangled representations." In: *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2525–2534 (cit. on p. 95).

Esmaeili, Babak, Hao Wu, Sarthak Jain, Alican Bozkurt, N Siddharth, Brooks Paige, Dana H. Brooks, Jennifer Dy, and Jan-Willem van de Meent (Apr. 2019b). "Structured Disentangled Representations." In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 2525–2534 (cit. on p. 113).

Evans, Richard and Edward Grefenstette (2018). "Learning explanatory rules from noisy data." In: *Journal of Artificial Intelligence Research* 61, pp. 1–64 (cit. on p. 30).

Fawzi, Alhussein and Pascal Frossard (2016). "Measuring the effect of nuisance variables on classifiers." In: *British Machine Vision Conference (BMVC)*. CONF (cit. on p. 6).

Fidler, Sanja, Sven Dickinson, and Raquel Urtasun (2012). "3d object detection and viewpoint estimation with a deformable 3d cuboid model." In: *Advances in neural information processing systems*, pp. 611–619 (cit. on pp. 38, 66).

Fikes, Richard E and Nils J Nilsson (1971). "STRIPS: A new approach to the application of theorem proving to problem solving." In: *Artificial intelligence* 2.3-4, pp. 189–208 (cit. on p. 29).

Finn, Chelsea, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel (2016). "Deep spatial autoencoders for visuomotor learning." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 512–519 (cit. on pp. 7, 9, 80).

Finn, Chelsea, Tianhe Yu, Justin Fu, Pieter Abbeel, and Sergey Levine (2017). "Generalizing skills with semi-supervised reinforcement learning." In: *International Conference on Learning Representations* (cit. on p. 94).

Földiák, Peter (1991). "Learning invariance from transformation sequences." In: *Neural Computation* 3.2, pp. 194–200 (cit. on p. 26).

Fowlkes, Edward B and Colin L Mallows (1983). "A method for comparing two hierarchical clusterings." In: *Journal of the American statistical association* 78.383, pp. 553–569 (cit. on pp. 35, 158).

Freeman, William T and Michal Roth (1995). "Orientation histograms for hand gesture recognition." In: *International workshop on automatic face and gesture recognition*. Vol. 12. Zurich, Switzerland, pp. 296–301 (cit. on p. 6).

Freeman, William T, Ken-ichi Tanaka, Jun Ohta, and Kazuo Kyuma (1996). "Computer vision for computer games." In: *Proceedings of the second international conference on automatic face and gesture recognition*. IEEE, pp. 100–105 (cit. on p. 6).

Fujimoto, Scott, Herke Hoof, and David Meger (2018). "Addressing function approximation error in actor-critic methods." In: *International Conference on Machine Learning*. PMLR, pp. 1587–1596 (cit. on pp. 83, 87).

Fukuda, Keisuke, Edward Awh, and Edward K Vogel (2010). "Discrete capacity limits in visual working memory." In: *Current opinion in neurobiology* 20.2, pp. 177–182 (cit. on p. 29).

Funk, Niklas, Charles Schaff, Rishabh Madan, Takuma Yoneda, Julen Urain De Jesus, Joe Watson, Ethan K. Gordon, Felix Widmaier, Stefan Bauer, Siddhartha S. Srinivasa, Tapomayukh Bhattacharjee, Matthew R. Walter, and Jan Peters (2021). "Benchmarking Structured Policies and Policy Optimization for Real-World Dexterous Object Manipulation." In: *IEEE Robotics and Automation Letters Special Issue: Robotic Grasping and Manipulation Challenges and Progress* (cit. on p. 80).

Gatys, Leon, Alexander Ecker, and Matthias Bethge (Aug. 2016). "A Neural Algorithm of Artistic Style." en. In: *Journal of Vision* 16.12. Publisher: The Association for Research in Vision and Ophthalmology, pp. 326–326. ISSN: 1534-7362. DOI: 10.1167/16.12.326 (cit. on pp. 105, 164).

Geirhos, Robert, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann (2020). "Shortcut learning in deep neural networks." In: *Nature Machine Intelligence* 2.11, pp. 665–673 (cit. on p. 13).

Gelernter, Herbert L (1959). "Realization of a geometry theorem proving machine." In: *IFIP congress*, pp. 273–281 (cit. on p. 29).

Gershman, Samuel and Noah Goodman (2014). "Amortized inference in probabilistic reasoning." In: *Proceedings of the annual meeting of the cognitive science society*. Vol. 36. 36 (cit. on p. 16).

Ghadirzadeh, Ali, Atsuto Maki, Danica Kragic, and Mårten Björkman (2017). "Deep predictive policy training using reinforcement learning." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2351–2358 (cit. on p. 81).

Ghallab, Malik, Dana Nau, and Paolo Traverso (2004). *Automated Planning: theory and practice*. Elsevier (cit. on p. 29).

Gidaris, Spyros, Praveer Singh, and Nikos Komodakis (2018). "Unsupervised representation learning by predicting image rotations." In: *arXiv preprint arXiv:1803.07728* (cit. on p. 8).

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik (June 2014). "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 8).

Glasserman, Paul (2004). *Monte Carlo methods in financial engineering*. Vol. 53. Springer (cit. on p. 17).

Gondal, Muhammad Waleed, Manuel Wüthrich, Djordje Miladinović, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer (2019). "On the Transfer of Inductive Bias from Simulation to the Real World: a New Disentanglement Dataset." In: *Advances in Neural Information Processing Systems* (cit. on pp. 38, 39, 66, 67, 95, 113).

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets." In: *Advances in neural information processing systems*, pp. 2672–2680 (cit. on p. 7).

Gordon, Robert D and David E Irwin (1996). "What's in an object file? Evidence from priming studies." In: *Perception & Psychophysics* 58.8, pp. 1260–1277 (cit. on p. 80).

Gowal, Sven, Chongli Qin, Po-Sen Huang, Taylan Cemgil, Krishnamurthy Dvijotham, Timothy Mann, and Pushmeet Kohli (2020). "Achieving robustness in the wild via adversarial mixing with disentangled representations." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1211–1220 (cit. on p. 67).

Goyal, Anirudh, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf (2019). "Recurrent independent mechanisms." In: *arXiv preprint arXiv:1909.10893* (cit. on p. 113).

Green, Edwin James (2019). "A theory of perceptual objects." In: *Philosophy and Phenomenological Research* 99.3, pp. 663–693 (cit. on p. 100).

Greff, Klaus, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner (2019). "Multi-object representation learning with iterative variational inference." In: *International Conference on Machine Learning*. PMLR, pp. 2424–2433 (cit. on pp. 54, 60, 98, 103, 104, 113, 145, 152, 153).

Greff, Klaus, Antti Rasmus, Mathias Berglund, Tele Hao, Harri Valpola, and Jürgen Schmidhuber (2016). "Tagger: Deep Unsupervised Perceptual Grouping." en. In: *Advances in Neural Information Processing Systems* 29. (Visited on 05/27/2021) (cit. on p. 113).

Greff, Klaus, Sjoerd van Steenkiste, and Jürgen Schmidhuber (2017). "Neural expectation maximization." In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6694–6704 (cit. on pp. 98, 113).

– (2020). "On the Binding Problem in Artificial Neural Networks." In: *arXiv preprint arXiv:2012.05208* (cit. on pp. 29–32, 51, 60, 98–101, 113).

Gregor, Karol, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra (2015). "Draw: A recurrent neural network for image generation." In: *International Conference on Machine Learning*. PMLR, pp. 1462–1471 (cit. on p. 98).

Gresele, Luigi, Paul K. Rubenstein, Arash Mehrjou, Francesco Locatello, and Bernhard Schölkopf (2019). "The Incomplete Rosetta Stone Problem: Identifiability Results for Multi-View Nonlinear ICA." In: *Conference on Uncertainty in Artificial Intelligence (UAI)* (cit. on p. 66).

Grill, Jean-Bastien, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. (2020). "Bootstrap your own latent-a new approach to self-supervised learning." In: *Advances in Neural Information Processing Systems* 33, pp. 21271–21284 (cit. on p. 9).

Groth, Oliver, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi (2018). "Shapestacks: Learning vision-based physical intuition for generalised object stacking." In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 702–717 (cit. on pp. 102, 152).

Gulrajani, Ishaan and David Lopez-Paz (2020). "In search of lost domain generalization." In: *arXiv preprint arXiv:2007.01434* (cit. on p. 80).

Ha, David and Jürgen Schmidhuber (2018). "Recurrent world models facilitate policy evolution." In: *arXiv preprint arXiv:1809.01999* (cit. on pp. 7, 81).

Haarnoja, Tuomas, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine (2018a). "Learning to walk via deep reinforcement learning." In: *arXiv preprint arXiv:1812.11103* (cit. on p. 87).

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, and Sergey Levine (2018b). "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor." In: *International Conference on Machine Learning*. PMLR, pp. 1861–1870 (cit. on pp. 47, 83).

Hamilton, William L (2020). "Graph representation learning." In: *Synthesis Lectures on Artifical Intelligence and Machine Learning* 14.3, pp. 1–159 (cit. on p. 8).

Harnad, Stevan (1990). "The symbol grounding problem." In: *Physica D: Nonlinear Phenomena* 42.1-3, pp. 335–346 (cit. on p. 29).

Håstad, Johan (1986). "Almost Optimal Lower Bounds for Small Depth Circuits." In: *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*. Ed. by Juris Hartmanis. ACM, pp. 6–20 (cit. on p. 7).

Håstad, Johan and Mikael Goldmann (1991). "On the power of small-depth threshold circuits." In: *Computational Complexity* 1.2, pp. 113–129 (cit. on p. 7).

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick (2020). "Momentum contrast for unsupervised visual representation learning." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738 (cit. on p. 8).

Heaven, Will Douglas (Jan. 5, 2021). "This avocado armchair could be the future of AI." In: *MIT Technology Review*. URL: https://www.technologyreview.com/2021/01/05/1015754/avocado-armchair-future-ai-openai-deep-learning-nlp-gpt3-computer-vision-common-sense/ (cit. on p. 13).

Heinze-Deml, Christina and Nicolai Meinshausen (2017). "Conditional variance penalties and domain shift robustness." In: *arXiv preprint arXiv:1710.11469* (cit. on p. 67).

Hénaff, Olivier J, Skanda Koppula, Jean-Baptiste Alayrac, Aaron van den Oord, Oriol Vinyals, and João Carreira (2021). "Efficient visual pretraining with contrastive detection." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10086–10096 (cit. on p. 9).

Hendrycks, Dan and Thomas Dietterich (2019). "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations." In: *Proceedings of the International Conference on Learning Representations* (cit. on p. 80).

Hendrycks, Dan, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song (2021). "Natural adversarial examples." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271 (cit. on p. 12).

Henighan, Tom, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. (2020). "Scaling laws for autoregressive generative modeling." In: *arXiv preprint arXiv:2010.14701* (cit. on p. 13).

Higgins, Irina, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner (2017a). "beta-VAE: Learning basic visual concepts with a constrained variational framework." In: *International Conference on Learning Representations* (cit. on pp. 18, 19, 23, 39, 66, 82, 86, 100).

Higgins, Irina, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner (2017b). "DARLA: Improving Zero-Shot Transfer in Reinforcement Learning." In: *International Conference on Machine Learning* (cit. on pp. 64, 67, 95).

Higgins, Irina, Nicolas Sonnerat, Loic Matthey, Arka Pal, Christopher P Burgess, Matko Bošnjak, Murray Shanahan, Matthew Botvinick, Demis Hassabis, and Alexander Lerchner (2018). "SCAN: Learning Hierarchical Compositional Visual Concepts." In: *International Conference on Learning Representations* (cit. on pp. 11, 64).

Hill, Ashley, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu (2018). *Stable Baselines*. https://github.com/hill-a/stable-baselines (cit. on p. 136).

Hinton, Geoffrey, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury (2012). "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups." In: *IEEE Signal Processing Magazine* 29.6, pp. 82–97. DOI: 10.1109/MSP.2012.2205597 (cit. on p. 7).

Hinton, Geoffrey E and Ruslan R Salakhutdinov (2006). "Reducing the dimensionality of data with neural networks." In: *Science* 313.5786, pp. 504–507 (cit. on p. 8).

Hirsh, Haym and Nathalie Japkowicz (1994). "Bootstrapping training-data representations for inductive learning: A case study in molecular biology." In: *AAAI*. Citeseer, pp. 639–644 (cit. on p. 6).

Hosoya, Haruo (2019). "Group-based Learning of Disentangled Representations with Generalizability for Novel Contents." In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by Sarit Kraus, pp. 2506–2513 (cit. on pp. 26, 27).

Huang, Qian, Horace He, Abhay Singh, Yan Zhang, Ser Nam Lim, and Austin R Benson (2020). "Better set representations for relational reasoning." In: *Advances in Neural Information Processing Systems* 33 (cit. on p. 113).

Hubert, Lawrence and Phipps Arabi (Dec. 1985). "Comparing partitions." en. In: *Journal of Classification* 2.1, pp. 193–218. ISSN: 1432-1343. DOI: 10.1007/BF01908075 (cit. on pp. 34, 35, 103, 157, 158).

Hyvarinen, Aapo and Hiroshi Morioka (2016). "Unsupervised feature extraction by time-contrastive learning and nonlinear ICA." In: *Advances in Neural Information Processing Systems* (cit. on p. 66).

Hyvarinen, Aapo, Hiroaki Sasaki, and Richard E Turner (2019). "Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning." In: *International Conference on Artificial Intelligence and Statistics* (cit. on p. 66).

Hyvärinen, Aapo and Petteri Pajunen (1999). "Nonlinear independent component analysis: Existence and uniqueness results." In: *Neural Networks* (cit. on p. 66).

Ilyas, Andrew, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry (2019). "Adversarial examples are not bugs, they are features." In: *Advances in neural information processing systems* 32 (cit. on p. 12).

Jacq, Alexis and Winston Herring (2021). *Neural Transfer Using PyTorch*. en. URL: https://github.com/pytorch/tutorials (visited on 05/26/2021) (cit. on p. 164).

James, Stephen, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis (2019). "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12627–12637 (cit. on pp. 67, 95).

Jean, Sébastien, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio (2014). "On using very large target vocabulary for neural machine translation." In: *arXiv preprint arXiv:1412.2007* (cit. on p. 7).

Jiang*, Jindong, Sepehr Janghorbani*, Gerard De Melo, and Sungjin Ahn (2020). "SCALOR: Generative World Models with Scalable Object Representations." In: *International Conference on Learning Representations* (cit. on p. 114).

Jo, Jason and Yoshua Bengio (2017). "Measuring the tendency of cnns to learn surface statistical regularities." In: *arXiv preprint arXiv:1711.11561* (cit. on p. 12).

John, George H, Ron Kohavi, and Karl Pfleger (1994). "Irrelevant features and the subset selection problem." In: *Machine learning proceedings 1994*. Elsevier, pp. 121–129 (cit. on p. 5).

Johnson, Justin, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick (2017). "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2901–2910 (cit. on pp. 102, 152).

Joulin, Armand, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache (2016). "Learning Visual Features from Large Weakly Supervised Data." In: *Computer Vi-*

sion – ECCV 2016. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Cham: Springer International Publishing, pp. 67–84 (cit. on p. 8).

Jutten, Christian and Juha Karhunen (2003). "Advances in nonlinear blind source separation." In: *International Symposium on Independent Component Analysis and Blind Signal Separation*, pp. 245–256 (cit. on p. 66).

Kabra, Rishabh, Chris Burgess, Loic Matthey, Raphael Lopez Kaufman, Klaus Greff, Malcolm Reynolds, and Alexander Lerchner (2019). *Multi-Object Datasets*. URL: https://github.com/deepmind/multi-object-datasets/ (cit. on pp. 55, 102, 152).

Kabra, Rishabh, Daniel Zoran, Goker Erdogan, Loic Matthey, Antonia Creswell, Matt Botvinick, Alexander Lerchner, and Chris Burgess (2021). "Simone: View-invariant, temporally-abstracted object representations via unsupervised video decomposition." In: *Advances in Neural Information Processing Systems* 34 (cit. on p. 32).

Kaiser, Lukasz, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. (2019). "Model-based reinforcement learning for atari." In: *arXiv preprint arXiv:1903.00374* (cit. on p. 80).

Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei (2020). "Scaling laws for neural language models." In: *arXiv preprint arXiv:2001.08361* (cit. on p. 13).

Ke, Nan Rosemary, Aniket Rajiv Didolkar, Sarthak Mittal, Anirudh Goyal, Guillaume Lajoie, Stefan Bauer, Danilo Jimenez Rezende, Michael Curtis Mozer, Yoshua Bengio, and Christopher Pal (2021). *Systematic Evaluation of Causal Discovery in Visual Model Based Reinforcement Learning*. URL: https://openreview.net/forum?id=gp5Uzbl-9C- (cit. on p. 80).

Khemakhem, Ilyes, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen (2020). "Variational autoencoders and nonlinear ica: A unifying framework." In: *International Conference on Artificial Intelligence and Statistics*, pp. 2207–2217 (cit. on pp. 26, 66).

Khurana, Udayan, Horst Samulowitz, and Deepak Turaga (2018). "Feature engineering for predictive modeling using reinforcement learning." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1 (cit. on p. 6).

Khurana, Udayan, Deepak Turaga, Horst Samulowitz, and Srinivasan Parthasrathy (2016). "Cognito: Automated feature engineering for supervised learning." In: *2016*

IEEE 16th International Conference on Data Mining Workshops (ICDMW). IEEE, pp. 1304–1307 (cit. on p. 6).

Kibbe, Melissa M and Alan M Leslie (2019). "Conceptually rich, perceptually sparse: Object representations in 6-month-old infants' working memory." In: *Psychological Science* 30.3, pp. 362–375 (cit. on p. 29).

Kim, Hyunjik and Andriy Mnih (2018). "Disentangling by factorising." In: *International Conference on Machine Learning* (cit. on pp. 20, 21, 23, 39, 66, 86, 100).

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980* (cit. on p. 123).

Kingma, Diederik P, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling (2014). "Semi-supervised learning with deep generative models." In: *Advances in Neural Information Processing Systems* (cit. on p. 7).

Kingma, Diederik P and Max Welling (2014). "Auto-encoding variational Bayes." In: *International Conference on Learning Representations* (cit. on pp. 16, 17, 65, 82, 102, 145).

Kipf, Thomas, Gamaleldin F Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff (2021). "Conditional Object-Centric Learning from Video." In: *arXiv preprint arXiv:2111.12594* (cit. on pp. 32, 113).

Kipf, Thomas, Elise van der Pol, and Max Welling (2019). "Contrastive learning of structured world models." In: *arXiv preprint arXiv:1911.12247* (cit. on pp. 32, 113).

Kiran, B Ravi, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez (2021). "Deep reinforcement learning for autonomous driving: A survey." In: *IEEE Transactions on Intelligent Transportation Systems* (cit. on p. 87).

Klindt, David, Lukas Schott, Yash Sharma, Ivan Ustyuzhaninov, Wieland Brendel, Matthias Bethge, and Dylan Paiton (2020). "Towards Nonlinear Disentanglement in Natural Data with Temporal Sparse Coding." In: *arXiv preprint arXiv:2007.10930* (cit. on p. 66).

Klys, Jack, Jake Snell, and Richard Zemel (2018). "Learning Latent Subspaces in Variational Autoencoders." In: *Advances in Neural Information Processing Systems* (cit. on p. 26).

Koh, Pang Wei, Shiori Sagawa, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, et al. (2021). "Wilds: A benchmark of in-the-wild distribution shifts." In: *International Conference on Machine Learning*. PMLR, pp. 5637–5664 (cit. on p. 80).

Kolesnikov, Alexander, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby (2020). "Big Transfer (BiT): General Visual Representation Learning." In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm. Springer International Publishing, pp. 491–507 (cit. on p. 8).

Kolesnikov, Alexander, Xiaohua Zhai, and Lucas Beyer (2019). "Revisiting self-supervised visual representation learning." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1920–1929 (cit. on p. 8).

Kormushev, Petar, Sylvain Calinon, and Darwin G Caldwell (2013). "Reinforcement learning in robotics: Applications and real-world challenges." In: *Robotics* 2.3, pp. 122–148 (cit. on p. 95).

Kosiorek, Adam R, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh (2018). "Sequential attend, infer, repeat: Generative modelling of moving objects." In: *arXiv preprint arXiv:1806.01794* (cit. on pp. 98, 113).

Kossen, Jannik, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting (2019). "Structured Object-Aware Physics Prediction for Video Modeling and Planning." In: *arXiv preprint arXiv:1910.02425* (cit. on p. 113).

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks." In: *Advances in neural information processing systems* 25 (cit. on p. 7).

Krueger, David, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Remi Le Priol, and Aaron Courville (2020). "Out-of-distribution generalization via risk extrapolation (rex)." In: *arXiv preprint arXiv:2003.00688* (cit. on p. 67).

Kuhn, Harold W (1955). "The Hungarian method for the assignment problem." In: *Naval research logistics quarterly* 2.1-2, pp. 83–97 (cit. on pp. 101, 103).

Kulkarni, Tejas D, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum (2015). "Deep convolutional inverse graphics network." In: *Advances in neural information processing systems* 28 (cit. on pp. 12, 18, 95).

Kullback, Solomon and Richard A Leibler (1951). "On information and sufficiency." In: *The annals of mathematical statistics* 22.1, pp. 79–86 (cit. on p. 15).

Kumar, Abhishek, Prasanna Sattigeri, and Avinash Balakrishnan (2018). "Variational inference of disentangled latent concepts from unlabeled observations." In: *International Conference on Learning Representations* (cit. on pp. 21, 25, 66, 92, 100).

Lake, Brenden M, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman (2017). "Building machines that learn and think like people." In: *Behavioral and Brain Sciences* 40 (cit. on pp. 18, 29, 51, 80, 98).

Lam, Hoang Thanh, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, and Oznur Alkan (2017). "One button machine for automating feature engineering in relational databases." In: *arXiv preprint arXiv:1706.00327* (cit. on p. 6).

Laskin, Michael, Aravind Srinivas, and Pieter Abbeel (2020). "Curl: Contrastive unsupervised representations for reinforcement learning." In: *International Conference on Machine Learning*. PMLR, pp. 5639–5650 (cit. on p. 9).

Le, Hai-Son, Ilya Oparin, Alexandre Allauzen, Jean-Luc Gauvain, and François Yvon (2012). "Structured output layer neural network language models for speech recognition." In: *IEEE Transactions on Audio, Speech, and Language Processing* 21.1, pp. 197–206 (cit. on p. 7).

Le Roux, Nicolas, Nicolas Heess, Jamie Shotton, and John Winn (2011). "Learning a generative model of images by factoring appearance and shape." In: *Neural Computation* 23.3, pp. 593–650 (cit. on p. 113).

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning." In: *Nature* 521.7553, p. 436 (cit. on p. 6).

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (1998). "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11, pp. 2278–2324 (cit. on p. 6).

LeCun, Yann, Fu Jie Huang, and Leon Bottou (2004). "Learning methods for generic object recognition with invariance to pose and lighting." In: *IEEE Conference on Computer Vision and Pattern Recognition* (cit. on pp. 39, 66).

Lee, Hsin-Ying, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang (2018). "Diverse image-to-image translation via disentangled representations." In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 35–51 (cit. on p. 19).

Lee, Youngwoon, Edward S Hu, and Joseph J Lim (2021). "IKEA furniture assembly environment for long-horizon complex manipulation tasks." In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6343–6349 (cit. on p. 9).

Lesort, Timothée, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat (2018). "State representation learning for control: An overview." In: *Neural Networks* 108, pp. 379–392 (cit. on p. 8).

Li, Ya, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao (2018). "Deep domain generalization via conditional invariant adversarial networks." In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639 (cit. on p. 67).

Liévin, Valentin, Andrea Dittadi, Anders Christensen, and Ole Winther (2020). "Optimal Variance Control of the Score-Function Gradient Estimator for Importance-Weighted Bounds." In: *Advances in Neural Information Processing Systems* 33, pp. 16591–16602 (cit. on pp. xv, 17).

Liévin, Valentin, Andrea Dittadi, Lars Maaløe, and Ole Winther (2019). "Towards hierarchical discrete variational autoencoders." In: *2nd Symposium on Advances in Approximate Bayesian Inference* (cit. on p. xv).

Lillicrap, Timothy P, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra (2015). "Continuous control with deep reinforcement learning." In: *arXiv preprint arXiv:1509.02971* (cit. on p. 7).

Lin, Zhixuan, Yi-Fu Wu, Skand Peri, Bofeng Fu, Jindong Jiang, and Sungjin Ahn (2020a). "Improving Generative Imagination in Object-Centric World Models." In: *International Conference on Machine Learning*. PMLR, pp. 6140–6149 (cit. on p. 114).

Lin, Zhixuan, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn (2020b). "SPACE: Unsupervised Object-Oriented Scene Representation via Spatial Attention and Decomposition." In: *ICLR 2020* (cit. on pp. 34, 98, 102, 114, 144, 147, 148).

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). "Deep learning face attributes in the wild." In: *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738 (cit. on p. 39).

Locatello, Francesco, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem (2019a). "On the fairness of disentangled representations." In: *Advances in Neural Information Processing Systems*, pp. 14611–14624 (cit. on pp. 11, 18, 64, 77, 91, 113).

Locatello, Francesco, Stefan Bauer, Mario Lucic, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem (2019b). "Challenging common assumptions in the unsupervised learning of disentangled representations." In: *International Conference on Machine Learning* (cit. on pp. 19, 25, 66, 68, 70, 74, 100, 113, 123, 125).

Locatello, Francesco, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem (2020a). "A Sober Look at the Unsupervised Learning of Disentangled Representations and their Evaluation." In: *Journal of Machine Learning Research* 21, pp. 1–62 (cit. on pp. 22–25, 40).

Locatello, Francesco, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen (2020b). "Weakly-Supervised Disentanglement Without Compromises." In: *arXiv preprint arXiv:2002.02886* (cit. on pp. 11, 26–28, 40, 66, 67, 69, 71, 82, 86, 123).

Locatello, Francesco, Michael Tschannen, Stefan Bauer, Gunnar Rätsch, Bernhard Schölkopf, and Olivier Bachem (2020c). "Disentangling Factors of Variations Using Few Labels." In: *8th International Conference on Learning Representations, ICLR 2020* (cit. on pp. 22, 26).

Locatello, Francesco, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf (2020d). "Object-centric learning with slot attention." In: *arXiv preprint arXiv:2006.15055* (cit. on pp. 32–34, 98, 101–103, 113, 144, 146, 152, 153, 157, 192, 193).

Lowe, David G (1999). "Object recognition from local scale-invariant features." In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee, pp. 1150–1157 (cit. on p. 6).

Löwe, Sindy, Klaus Greff, Rico Jonschkowski, Alexey Dosovitskiy, and Thomas Kipf (2020). "Learning object-centric video models by contrasting sets." In: *arXiv preprint arXiv:2011.10287* (cit. on pp. 32, 113).

Makhzani, Alireza, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey (2015). "Adversarial autoencoders." In: *arXiv preprint arXiv:1511.05644* (cit. on p. 20).

Mao, Jiayuan, Chuang Gan, Pushmeet Kohli, Joshua B. Tenenbaum, and Jiajun Wu (2019). "The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision." In: *International Conference on Learning Representations* (cit. on p. 30).

Marcus, Gary (2018). "Deep learning: A critical appraisal." In: *arXiv preprint arXiv:1801.00631* (cit. on pp. 29, 30).

– (Mar. 10, 2022). "Deep Learning is Hitting a Wall." In: *Nautilus*. URL: https://nautil.us/deep-learning-is-hitting-a-wall-14467/ (cit. on p. 13).

Marcus, Gary F (2003). *The algebraic mind: Integrating connectionism and cognitive science*. MIT press (cit. on p. 29).

Markovitch, Shaul and Dan Rosenstein (2002). "Feature generation using general constructor functions." In: *Machine Learning* 49.1, pp. 59–98 (cit. on p. 6).

Marr, David (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc. (cit. on p. 80).

Matheus, Christopher J and Larry A Rendell (1989). "Constructive Induction On Decision Trees." In: *IJCAI*. Vol. 89. Citeseer, pp. 645–650 (cit. on p. 6).

Mathieu, Michael F, Junbo J Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun (2016). "Disentangling factors of variation in deep representation using ad-

versarial training." In: *Advances in Neural Information Processing Systems* (cit. on p. 19).

Matthey, Loic, Irina Higgins, Demis Hassabis, and Alexander Lerchner (2017). *dSprites: Disentanglement testing Sprites dataset*. URL: https://github.com/deepmind/dsprites-dataset/ (cit. on p. 153).

Meehl, Paul E (1970). *Nuisance variables and the ex post facto design*. University of Minnesota Press, Minneapolis (cit. on p. 6).

Mehrabi, Ninareh, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan (2021). "A survey on bias and fairness in machine learning." In: *ACM Computing Surveys (CSUR)* 54.6, pp. 1–35 (cit. on pp. 12, 13).

Michaelis, Claudio, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel (2019). "Benchmarking robustness in object detection: Autonomous driving when winter is coming." In: *arXiv preprint 1907.07484* (cit. on p. 80).

Mikolov, Tomáš, Anoop Deoras, Daniel Povey, Lukáš Burget, and Jan Černockỳ (2011). "Strategies for training large scale neural network language models." In: *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, pp. 196–201 (cit. on p. 7).

Miller, George A (1956). "The magical number seven, plus or minus two: Some limits on our capacity for processing information." In: *Psychological review* 63.2, p. 81 (cit. on p. 29).

Mirza, Mehdi and Simon Osindero (2014). "Conditional generative adversarial nets." In: *arXiv preprint arXiv:1411.1784* (cit. on p. 7).

Mitchell, Tom M (1980). *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research … (cit. on p. 13).

Mnih, Andriy and Karol Gregor (2014). "Neural variational inference and learning in belief networks." In: *International Conference on Machine Learning*. PMLR, pp. 1791–1799 (cit. on p. 17).

Mnih, Andriy and Danilo Rezende (2016). "Variational inference for monte carlo objectives." In: *International Conference on Machine Learning*. PMLR, pp. 2188–2196 (cit. on p. 17).

Mnih, Volodymyr, Nicolas Heess, Alex Graves, et al. (2014). "Recurrent models of visual attention." In: *Advances in neural information processing systems*, pp. 2204–2212 (cit. on p. 98).

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg

Ostrovski, et al. (2015). "Human-level control through deep reinforcement learning." In: *nature* 518.7540, pp. 529–533 (cit. on p. 7).

Mohamed, Abdel-rahman, George E Dahl, and Geoffrey Hinton (2011). "Acoustic modeling using deep belief networks." In: *IEEE transactions on audio, speech, and language processing* 20.1, pp. 14–22 (cit. on p. 7).

Mohamed, Shakir, Mihaela Rosca, Michael Figurnov, and Andriy Mnih (2020). "Monte Carlo Gradient Estimation in Machine Learning." In: *J. Mach. Learn. Res.* 21.132, pp. 1–62 (cit. on p. 46).

Montero, Milton Llera, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers (2021). "The role of Disentanglement in Generalisation." In: *International Conference on Learning Representations* (cit. on p. 113).

Montufar, Guido F, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio (2014). "On the number of linear regions of deep neural networks." In: *Advances in neural information processing systems* 27 (cit. on p. 7).

Mourao, Kira, Luke S Zettlemoyer, Ronald Petrick, and Mark Steedman (2012). "Learning STRIPS operators from noisy and incomplete observations." In: *arXiv preprint arXiv:1210.4889* (cit. on p. 29).

Muandet, Krikamol, David Balduzzi, and Bernhard Schölkopf (2013). "Domain generalization via invariant feature representation." In: *International Conference on Machine Learning*, pp. 10–18 (cit. on p. 67).

Murphy, Kevin P (2012). *Machine learning: a probabilistic perspective*. MIT press (cit. on pp. 5, 6).

Nair, Ashvin V, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine (2018). "Visual reinforcement learning with imagined goals." In: *Advances in Neural Information Processing Systems* (cit. on p. 81).

Nargesian, Fatemeh, Horst Samulowitz, Udayan Khurana, Elias B Khalil, and Deepak S Turaga (2017). "Learning Feature Engineering for Classification." In: *Ijcai*, pp. 2529–2535 (cit. on p. 6).

Nash, Charlie, SM Ali Eslami, Chris Burgess, Irina Higgins, Daniel Zoran, Theophane Weber, and Peter Battaglia (2017). "The multi-entity variational autoencoder." In: *Neural Information Processing Systems (NeurIPS) Workshop on Learning Disentangled Representations: from Perception to Control* (cit. on p. 114).

Newell, Allen and Herbert Simon (1956). "The logic theory machine–A complex information processing system." In: *IRE Transactions on information theory* 2.3, pp. 61–79 (cit. on p. 29).

Oberauer, Klaus (2019). "Working memory and attention—A conceptual analysis and review." In: *Journal of cognition* (cit. on p. 29).

Oord, Aaron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu (2016). "Wavenet: A generative model for raw audio." In: *arXiv preprint arXiv:1609.03499* (cit. on p. 7).

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals (2018). "Representation learning with contrastive predictive coding." In: *arXiv preprint arXiv:1807.03748* (cit. on p. 8).

Oppenheim, Alan V. and Ronald W. Schafer (2009). *Discrete-Time Signal Processing*. 3rd. USA: Prentice Hall Press. ISBN: 0131988425 (cit. on p. 6).

Paige, Brooks, Jan-Willem van de Meent, Alban Desmaison, Noah Goodman, Pushmeet Kohli, Frank Wood, Philip Torr, et al. (2017). "Learning disentangled representations with semi-supervised deep generative models." In: *Advances in neural information processing systems* 30 (cit. on p. 26).

Pálsson, Sveinn, Stefano Cerri, Andrea Dittadi, and Koen Van Leemput (2019). "Semi-supervised variational autoencoder for survival prediction." In: *International MICCAI Brainlesion Workshop*. Springer, pp. 124–134 (cit. on p. xv).

Pan, Sinno Jialin, Ivor W Tsang, James T Kwok, and Qiang Yang (2010). "Domain adaptation via transfer component analysis." In: *IEEE transactions on neural networks* 22.2, pp. 199–210 (cit. on p. 8).

Pan, Sinno Jialin and Qiang Yang (2009). "A survey on transfer learning." In: *IEEE Transactions on knowledge and data engineering* 22.10, pp. 1345–1359 (cit. on p. 8).

Papa, Samuele, Ole Winther, and Andrea Dittadi (2022). "Inductive Biases for Object-Centric Representations in the Presence of Complex Textures." In: *arXiv preprint arXiv:2204.08479* (cit. on pp. xiv, 2, 9, 56, 57, 109, 193).

Pascanu, Razvan, Guido Montufar, and Yoshua Bengio (2013). "On the number of response regions of deep feed forward networks with piece-wise linear activations." In: *arXiv preprint arXiv:1312.6098* (cit. on p. 7).

Pasula, Hanna M, Luke S Zettlemoyer, and Leslie Pack Kaelbling (2007). "Learning symbolic models of stochastic domains." In: *Journal of Artificial Intelligence Research* 29, pp. 309–352 (cit. on p. 29).

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. (2019). "Pytorch: An imperative style, high-performance deep learning library." In: *arXiv preprint arXiv:1912.01703* (cit. on pp. 55, 145).

Pearl, Judea (2009). *Causality*. Cambridge University Press (cit. on pp. 25, 98).

– (2018). "Theoretical impediments to machine learning with seven sparks from the causal revolution." In: *arXiv preprint arXiv:1801.04016* (cit. on p. 29).

Pei, Zhongyi, Zhangjie Cao, Mingsheng Long, and Jianmin Wang (2018). "Multi-adversarial domain adaptation." In: *Thirty-second AAAI conference on artificial intelligence* (cit. on p. 12).

Peng, Xue Bin, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel (2018). "Sim-to-real transfer of robotic control with dynamics randomization." In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 1–8 (cit. on p. 67).

Peters, Jonas, Dominik Janzing, and Bernhard Schölkopf (2017). *Elements of causal inference: foundations and learning algorithms*. The MIT Press (cit. on pp. 18, 98).

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018). "Deep Contextualized Word Representations." In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1*. Ed. by Marilyn A. Walker, Heng Ji, and Amanda Stent. Association for Computational Linguistics, pp. 2227–2237 (cit. on p. 7).

Pfister, Niklas, Stefan Bauer, and Jonas Peters (2019). "Learning stable and predictive structures in kinetic systems." In: *Proceedings of the National Academy of Sciences* 116.51, pp. 25405–25411 (cit. on p. 80).

Pollack, Jordan B (1990). "Recursive distributed representations." In: *Artificial Intelligence* 46.1-2, pp. 77–105 (cit. on p. 30).

Pratt, Lorien and Barbara Jennings (1996). "A survey of transfer between connectionist networks." In: *Connection Science* 8.2, pp. 163–184 (cit. on p. 8).

Rabiner, Lawrence R and Bernard Gold (1975). "Theory and application of digital signal processing." In: *Englewood Cliffs: Prentice-Hall* (cit. on p. 6).

Racah, Evan and Sarath Chandar (2020). "Slot Contrastive Networks: A Contrastive Approach for Representing Objects." In: *arXiv preprint arXiv:2007.09294* (cit. on p. 113).

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. (2021a). "Learning transferable visual models from natural language supervision." In: *International Conference on Machine Learning*. PMLR, pp. 8748–8763 (cit. on p. 13).

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever (2021b). "Learning Transferable Visual Models From Natural Language Supervision." In: *Proceedings of the 38th International Confer-*

ence on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 8748–8763 (cit. on p. 9).

Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). "Improving language understanding by generative pre-training." In: (cit. on p. 8).

Ragavan, Harish, Larry Rendell, Michael Shaw, and Antoinette Tessmer (1993). "Complex concept acquisition through directed search and feature caching." In: *IJCAI*. Citeseer, pp. 946–951 (cit. on p. 5).

Raghu, Maithra, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein (Aug. 2017). "On the Expressive Power of Deep Neural Networks." In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 2847–2854 (cit. on p. 7).

Rainforth, Tom, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh (2018). "Tighter variational bounds are not necessarily better." In: *International Conference on Machine Learning*. PMLR, pp. 4277–4285 (cit. on p. 17).

Ramesh, Aditya, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen (2022). *Hierarchical Text-Conditional Image Generation with CLIP Latents* (cit. on p. 13).

Ramesh, Aditya, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever (2021). "Zero-shot text-to-image generation." In: *International Conference on Machine Learning*. PMLR, pp. 8821–8831 (cit. on p. 13).

Rand, William M (1971). "Objective criteria for the evaluation of clustering methods." In: *Journal of the American Statistical association* 66.336, pp. 846–850 (cit. on pp. 35, 158).

Ranganath, Rajesh, Sean Gerrish, and David Blei (2014). "Black box variational inference." In: *Artificial intelligence and statistics*. PMLR, pp. 814–822 (cit. on p. 17).

Ranzato, Marc'Aurelio, Fu Jie Huang, Y-Lan Boureau, and Yann LeCun (2007). "Unsupervised learning of invariant feature hierarchies with applications to object recognition." In: *2007 IEEE conference on computer vision and pattern recognition*. IEEE, pp. 1–8 (cit. on p. 8).

Ranzato, Marc'Aurelio, Christopher Poultney, Sumit Chopra, and Yann Cun (2006). "Efficient learning of sparse representations with an energy-based model." In: *Advances in neural information processing systems* 19 (cit. on p. 8).

Reed, Scott, Kihyuk Sohn, Yuting Zhang, and Honglak Lee (2014). "Learning to disentangle factors of variation with manifold interaction." In: *International Conference on Machine Learning* (cit. on p. 26).

Reed, Scott, Yi Zhang, Yuting Zhang, and Honglak Lee (2015). "Deep visual analogy-making." In: *Advances in Neural Information Processing Systems* (cit. on pp. 39, 66).

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic backpropagation and approximate inference in deep generative models." In: *arXiv preprint arXiv:1401.4082* (cit. on pp. 16, 65, 82, 102, 145).

Rezende, Danilo Jimenez and Fabio Viola (2018). "Taming vaes." In: *arXiv preprint arXiv:1810.00597* (cit. on pp. 33, 144).

Ridgeway, Karl and Michael C Mozer (2018). "Learning Deep Disentangled Embeddings with the F-Statistic Loss." In: *Advances in Neural Information Processing Systems* (cit. on pp. 24, 66, 92, 100).

Roeder, Geoffrey, Yuhuai Wu, and David K Duvenaud (2017). "Sticking the landing: Simple, lower-variance gradient estimators for variational inference." In: *Advances in Neural Information Processing Systems* 30 (cit. on p. 17).

Rojas-Carulla, Mateo, Bernhard Schölkopf, Richard Turner, and Jonas Peters (2018). "Invariant models for causal transfer learning." In: *The Journal of Machine Learning Research* 19.1, pp. 1309–1342 (cit. on p. 67).

Rolinek, Michal, Dominik Zietlow, and Georg Martius (2019). "Variational Autoencoders Recover PCA Directions (by Accident)." In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition* (cit. on p. 19).

Romijnders, Rob, Aravindh Mahendran, Michael Tschannen, Josip Djolonga, Marvin Ritter, Neil Houlsby, and Mario Lucic (2021). "Representation learning from videos in-the-wild: An object-centric approach." In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 177–187 (cit. on p. 101).

Roy, Prasun, Subhankar Ghosh, Saumik Bhattacharya, and Umapada Pal (2018). "Effects of degradations on deep neural network architectures." In: *arXiv preprint 1807.10108* (cit. on p. 80).

Ruder, Sebastian (2017). "An overview of multi-task learning in deep neural networks." In: *arXiv preprint arXiv:1706.05098* (cit. on p. 10).

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). "Learning representations by back-propagating errors." In: *nature* 323.6088, pp. 533–536 (cit. on p. 8).

Rusu, Andrei A, Matej Večerík, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell (2017). "Sim-to-real robot learning from pixels with progressive nets." In: *Conference on Robot Learning*, pp. 262–270 (cit. on pp. 67, 94).

Safran, Itay and Ohad Shamir (Aug. 2017). "Depth-Width Tradeoffs in Approximating Natural Functions with Neural Networks." In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 2979–2987 (cit. on p. 7).

Sainath, Tara N, Brian Kingsbury, Abdel-rahman Mohamed, George E Dahl, George Saon, Hagen Soltau, Tomas Beran, Aleksandr Y Aravkin, and Bhuvana Ramabhadran (2013). "Improvements to deep convolutional neural networks for LVCSR." In: *2013 IEEE workshop on automatic speech recognition and understanding*. IEEE, pp. 315–320 (cit. on p. 7).

Sanchez-Gonzalez, Alvaro, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia (Nov. 2020). "Learning to Simulate Complex Physics with Graph Networks." In: *International Conference on Machine Learning*. PMLR, pp. 8459–8468 (cit. on pp. 30, 98).

Schmidhuber, Jürgen (1992). "Learning factorial codes by predictability minimization." In: *Neural computation* 4.6, pp. 863–879 (cit. on p. 18).

– (2015). "Deep learning in neural networks: An overview." In: *Neural networks* 61, pp. 85–117 (cit. on p. 6).

Schölkopf, Bernhard (2019). "Causality for Machine Learning." In: *arXiv preprint arXiv:1911.10500* (cit. on pp. 26, 98).

Schölkopf, Bernhard, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio (2021). "Toward Causal Representation Learning." In: *Proceedings of the IEEE* 109.5, pp. 612–634 (cit. on pp. 11, 13, 18, 26, 29, 31, 67, 80, 98, 101).

Schulman, John, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz (2015a). "Trust region policy optimization." In: *International conference on machine learning*. PMLR, pp. 1889–1897 (cit. on pp. 7, 83).

Schulman, John, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel (2015b). "High-dimensional continuous control using generalized advantage estimation." In: *arXiv preprint arXiv:1506.02438* (cit. on p. 83).

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov (2017). "Proximal policy optimization algorithms." In: *arXiv preprint arXiv:1707.06347* (cit. on pp. 83, 87).

Schwenk, Holger, Anthony Rousseau, and Mohammed Attik (2012). "Large, pruned or continuous space language models on a gpu for statistical machine translation." In: *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pp. 11–19 (cit. on p. 7).

Scott, Sam and Stan Matwin (1999). "Feature engineering for text classification." In: *ICML*. Vol. 99. Citeseer, pp. 379–388 (cit. on p. 6).

Searle, John R (1980). "Minds, brains, and programs." In: *Behavioral and brain sciences* 3.3, pp. 417–424 (cit. on p. 29).

Seide, Frank, Gang Li, and Dong Yu (2011). "Conversational speech transcription using context-dependent deep neural networks." In: *Twelfth annual conference of the international speech communication association* (cit. on p. 7).

Sermanet, Pierre, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun (2013). "Overfeat: Integrated recognition, localization and detection using convolutional networks." In: *arXiv preprint arXiv:1312.6229* (cit. on p. 7).

Sharif Razavian, Ali, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson (June 2014). "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (cit. on p. 8).

Shu, Rui, Yining Chen, Abhishek Kumar, Stefano Ermon, and Ben Poole (2020). "Weakly Supervised Disentanglement with Guarantees." In: *8th International Conference on Learning Representations, ICLR 2020* (cit. on pp. 22, 26, 66).

Sietsma, Jocelyn and Robert JF Dow (1991). "Creating artificial neural networks that generalize." In: *Neural networks* 4.1, pp. 67–79 (cit. on pp. 69, 76).

Silver, David, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller (2014). "Deterministic policy gradient algorithms." In: *International conference on machine learning*. PMLR, pp. 387–395 (cit. on p. 83).

Simonyan, Karen and Andrew Zisserman (2014). "Two-stream convolutional networks for action recognition in videos." In: *Advances in neural information processing systems* 27 (cit. on p. 7).

Singh, Avi, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine (2019). "End-to-end robotic reinforcement learning without reward engineering." In: *arXiv preprint arXiv:1904.07854* (cit. on p. 87).

Smith, Brian Cantell (1998). *On the origin of objects*. MIT Press (cit. on p. 100).

Smolensky, Paul (1990). "Tensor product variable binding and the representation of symbolic structures in connectionist systems." In: *Artificial intelligence* 46.1-2, pp. 159–216 (cit. on p. 30).

Socher, Richard, Eric Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Ng (2011). "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection." In: *Advances in neural information processing systems* 24 (cit. on p. 7).

Sønderby, Casper Kaae, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther (2016). "Ladder variational autoencoders." In: *Advances in neural information processing systems*, pp. 3738–3746 (cit. on p. 69).

Sorrenson, Peter, Carsten Rother, and Ullrich Köthe (2020). "Disentanglement by Nonlinear ICA with General Incompressible-flow Networks (GIN)." In: *arXiv preprint arXiv:2001.04872* (cit. on pp. 26, 66).

Spelke, Elizabeth S (1990). "Principles of object perception." In: *Cognitive science* 14.1, pp. 29–56 (cit. on pp. 30, 80, 98).

Spelke, Elizabeth S and Katherine D Kinzler (2007). "Core knowledge." In: *Developmental science* 10.1, pp. 89–96 (cit. on pp. 30, 98).

Srinivas, Aravind, Michael Laskin, and Pieter Abbeel (2020). "Curl: Contrastive unsupervised representations for reinforcement learning." In: *arXiv preprint arXiv:2004.04136* (cit. on p. 80).

Steels, Luc (2008). "The symbol grounding problem has been solved. So what's next." In: *Symbols and embodiment: Debates on meaning and cognition*, pp. 223–244 (cit. on p. 29).

Steenbrugge, Xander, Sam Leroux, Tim Verbelen, and Bart Dhoedt (2018). "Improving Generalization for Abstract Reasoning Tasks Using Disentangled Feature Representations." In: *Workshop on Relational Representation Learning at NeurIPS* (cit. on p. 95).

Steenkiste, Sjoerd van, Michael Chang, Klaus Greff, and Jürgen Schmidhuber (2018). "Relational neural expectation maximization: Unsupervised discovery of objects and their interactions." In: *arXiv preprint arXiv:1802.10353* (cit. on p. 113).

Steenkiste, Sjoerd van, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly (2020). "Investigating object compositionality in generative adversarial networks." In: *Neural Networks* 130, pp. 309–325 (cit. on pp. 32, 113).

Steenkiste, Sjoerd van, Francesco Locatello, Jürgen Schmidhuber, and Olivier Bachem (2019). "Are disentangled representations helpful for abstract visual reasoning?" In: *Advances in Neural Information Processing Systems* 32 (cit. on pp. 11, 18, 64, 80, 113).

Stelzner, Karl, Robert Peharz, and Kristian Kersting (June 2019). "Faster Attend-Infer-Repeat with Tractable Probabilistic Models." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhudinov. Vol. 97. PMLR, pp. 5966–5975 (cit. on p. 113).

Stooke, Adam, Kimin Lee, Pieter Abbeel, and Michael Laskin (2021). "Decoupling representation learning from reinforcement learning." In: *International Conference on Machine Learning*. PMLR, pp. 9870–9879 (cit. on pp. 7, 9, 80).

Sun, Chen, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta (Oct. 2017). "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 8).

Suter, Raphael, Djordje Miladinovic, Bernhard Schölkopf, and Stefan Bauer (2019). "Robustly disentangled causal mechanisms: Validating deep representations for interventional robustness." In: *International Conference on Machine Learning*. PMLR, pp. 6056–6065 (cit. on pp. 25, 67).

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks." In: *Advances in neural information processing systems* 27 (cit. on p. 7).

Sutton, Richard (2019). "The bitter lesson." In: *Incomplete Ideas (blog)* 13, p. 12 (cit. on p. 13).

Sutton, Richard S and Christopher J Matheus (1991). "Learning polynomial functions by feature construction." In: *Machine Learning Proceedings 1991*. Elsevier, pp. 208–212 (cit. on p. 6).

Sutton, Richard S, David A McAllester, Satinder P Singh, Yishay Mansour, et al. (1999). "Policy gradient methods for reinforcement learning with function approximation." In: *NIPs*. Vol. 99. Citeseer, pp. 1057–1063 (cit. on p. 83).

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu (2018). "A survey on deep transfer learning." In: *International conference on artificial neural networks*. Springer, pp. 270–279 (cit. on p. 8).

Téglás, Ernő, Edward Vul, Vittorio Girotto, Michel Gonzalez, Joshua B. Tenenbaum, and Luca L. Bonatti (May 2011). "Pure Reasoning in 12-Month-Old Infants as Probabilistic Inference." In: *Science* 332, pp. 1054–1059. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1196404 (cit. on pp. 30, 98).

Telgarsky, Matus (June 2016). "Benefits of Depth in Neural Networks." In: *29th Annual Conference on Learning Theory*. Ed. by Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir. Vol. 49. Proceedings of Machine Learning Research. Columbia University, New York, New York, USA: PMLR, pp. 1517–1539 (cit. on p. 7).

Tian, Yonglong, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola (2020). "What makes for good views for contrastive learning?" In: *Advances in Neural Information Processing Systems* 33, pp. 6827–6839 (cit. on p. 8).

Tobin, Josh, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel (2017). "Domain randomization for transferring deep neural networks from simulation to the real world." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 23–30 (cit. on pp. 67, 94).

Torralba, Antonio and Alexei A Efros (2011). "Unbiased look at dataset bias." In: *CVPR 2011*. IEEE, pp. 1521–1528 (cit. on p. 12).

Träuble, Frederik, Elliot Creager, Niki Kilbertus, Francesco Locatello, Andrea Dittadi, Anirudh Goyal, Bernhard Schölkopf, and Stefan Bauer (2021). "On Disentangled Representations Learned From Correlated Data." In: *International Conference on Machine Learning*. PMLR, pp. 10401–10412 (cit. on pp. xv, 9, 11–13, 19, 26, 39, 60, 64, 67, 84, 86, 113).

Tschannen, Michael, Olivier Bachem, and Mario Lucic (2018). "Recent Advances in Autoencoder-Based Representation Learning." In: *arXiv preprint arXiv:1812.05069* (cit. on pp. 8, 18).

Tucker, George, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein (2017). "Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models." In: *Advances in Neural Information Processing Systems* 30 (cit. on p. 17).

Van Den Oord, Aäron, Sander Dieleman, and Benjamin Schrauwen (2014). "Transfer learning by supervised pre-training for audio-based music classification." In: *Conference of the International Society for Music Information Retrieval (ISMIR 2014)* (cit. on p. 8).

Van Engelen, Jesper E and Holger H Hoos (2020). "A survey on semi-supervised learning." In: *Machine Learning* 109.2, pp. 373–440 (cit. on p. 9).

Van Hoof, Herke, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters (2016). "Stable reinforcement learning with autoencoders for tactile and visual data." In: *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, pp. 3928–3934 (cit. on p. 81).

Vinyals, Oriol, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning." In: *Nature* 575.7782, pp. 350–354 (cit. on pp. 9, 30, 80, 98).

Viola, Paul, Michael J Jones, and Daniel Snow (2005). "Detecting pedestrians using patterns of motion and appearance." In: *International Journal of Computer Vision* 63.2, pp. 153–161 (cit. on p. 6).

Von Der Malsburg, Christoph (1986). "Am I thinking assemblies?" In: *Brain theory*. Springer, pp. 161–176 (cit. on pp. 31, 100).

Wagemans, Johan (2015). *The Oxford handbook of perceptual organization*. Oxford Library of Psychology (cit. on pp. 30, 98).

Wagner, Silke and Dorothea Wagner (2007). *Comparing clusterings: an overview*. Universität Karlsruhe, Fakultät für Informatik Karlsruhe (cit. on pp. 35, 158).

Walsh, Thomas J and Michael L Littman (2008). "Efficient learning of action schemas and web-service descriptions." In: *AAAI*. Vol. 8, pp. 714–719 (cit. on p. 29).

Wang, Jindong, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin (2021). "Generalizing to unseen domains: A survey on domain generalization." In: *arXiv preprint arXiv:2103.03097* (cit. on p. 95).

Wang, Mei and Weihong Deng (2018). "Deep visual domain adaptation: A survey." In: *Neurocomputing* 312, pp. 135–153 (cit. on p. 8).

Wang, Tongzhou and Phillip Isola (July 2020). "Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere." In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 9929–9939 (cit. on p. 8).

Watanabe, Satosi (1960). "Information theoretical analysis of multivariate correlation." In: *IBM Journal of research and development* 4.1, pp. 66–82 (cit. on p. 21).

Watter, Manuel, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller (2015). "Embed to control: A locally linear latent dynamics model for control from raw images." In: *arXiv preprint arXiv:1506.07365* (cit. on p. 81).

Watters, Nicholas, Loic Matthey, Christopher P Burgess, and Alexander Lerchner (2019). "Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes." In: *arXiv preprint arXiv:1901.07017* (cit. on pp. 102, 145).

Wegener, Ingo (1987). *The complexity of Boolean functions*. Wiley-Teubner. URL: http://ls2-www.cs.uni-dortmund.de/monographs/bluebook/ (cit. on p. 7).

Weis, Marissa A, Kashyap Chitta, Yash Sharma, Wieland Brendel, Matthias Bethge, Andreas Geiger, and Alexander S Ecker (2020). "Unmasking the Inductive Biases of Unsupervised Object Representations for Video Sequences." In: *arXiv preprint arXiv:2006.07034* (cit. on p. 98).

Williams, Ronald J (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning." In: *Machine learning* 8.3, pp. 229–256 (cit. on p. 17).

Wiskott, Laurenz and Terrence J Sejnowski (2002). "Slow feature analysis: Unsupervised learning of invariances." In: *Neural computation* 14.4, pp. 715–770 (cit. on p. 26).

Wu, Jiajun, Erika Lu, Pushmeet Kohli, Bill Freeman, and Josh Tenenbaum (2017). "Learning to see physics via visual de-animation." In: *Advances in Neural Information Processing Systems* 30 (cit. on p. 12).

Wu, Jiajun, Joshua B Tenenbaum, and Pushmeet Kohli (2017). "Neural scene de-rendering." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 699–707 (cit. on p. 12).

Wulfmeier, Markus, Arunkumar Byravan, Tim Hertweck, Irina Higgins, Ankush Gupta, Tejas Kulkarni, Malcolm Reynolds, Denis Teplyashin, Roland Hafner, Thomas Lampe, et al. (2021). "Representation matters: Improving perception and exploration for robotics." In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6512–6519 (cit. on p. 95).

Wüthrich, Manuel, Felix Widmaier, Felix Grimminger, Joel Akpo, Shruti Joshi, Vaibhav Agrawal, Bilal Hammoud, Majid Khadiv, Miroslav Bogdanovic, Vincent Berenz, et al. (2020). "TriFinger: An Open-Source Robot for Learning Dexterity." In: *arXiv preprint arXiv:2008.03596* (cit. on pp. 38, 64, 68, 81, 83, 85).

Xiao, Kai Yuanqing, Logan Engstrom, Andrew Ilyas, and Aleksander Madry (2021). "Noise or Signal: The Role of Image Backgrounds in Object Recognition." In: *International Conference on Learning Representations* (cit. on p. 12).

Yan, Mengyuan, Qingyun Sun, Iuri Frosio, Stephen Tyree, and Jan Kautz (2020). "How to Close Sim-Real Gap? Transfer with Segmentation!" In: *arXiv preprint arXiv:2005.07695* (cit. on p. 67).

Yang, Qiang, Kangheng Wu, and Yunfei Jiang (2007). "Learning action models from plan examples using weighted MAX-SAT." In: *Artificial Intelligence* 171.2-3, pp. 107–143 (cit. on p. 29).

Yang, Yanchao, Yutong Chen, and Stefano Soatto (2020). "Learning to manipulate individual objects in an image." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6558–6567 (cit. on p. 113).

Yi, Kexin, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum (2018). "Neural-symbolic vqa: Disentangling reasoning from vision and language understanding." In: *Advances in neural information processing systems* 31 (cit. on p. 30).

Yu, Tianhe, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine (2020). "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning." In: *Proceedings of the Conference on Robot Learning*. Ed. by Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura. Vol. 100. Proceedings of Machine Learning Research. PMLR, pp. 1094–1100 (cit. on p. 9).

Yuan, Jinyang, Bin Li, and Xiangyang Xue (2019). "Generative modeling of infinite occluded objects for compositional scene representation." In: *International Conference on Machine Learning*, pp. 7222–7231 (cit. on p. 98).

Zbontar, Jure, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny (July 2021). "Barlow Twins: Self-Supervised Learning via Redundancy Reduction." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 12310–12320 (cit. on p. 8).

Zeiler, Matthew D and Rob Fergus (2014). "Visualizing and understanding convolutional networks." In: *European conference on computer vision*. Springer, pp. 818–833 (cit. on pp. 7, 8).

Zhai, Xiaohua, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. (2019). "A large-scale study of representation learning with the visual task adaptation benchmark." In: *arXiv preprint arXiv:1910.04867* (cit. on p. 8).

Zhang, Amy, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine (2020). "Learning invariant representations for reinforcement learning without reconstruction." In: *arXiv preprint arXiv:2006.10742* (cit. on pp. 12, 80).

Zhang, Chiyuan, Oriol Vinyals, Remi Munos, and Samy Bengio (2018). "A study on overfitting in deep reinforcement learning." In: *arXiv preprint arXiv:1804.06893* (cit. on p. 80).

Zhao, Shengjia, Jiaming Song, and Stefano Ermon (2017). "Infovae: Information maximizing variational autoencoders." In: *arXiv preprint arXiv:1706.02262* (cit. on p. 20).

Zhu, Xiaojin (2005). *Semi-Supervised Learning Literature Survey*. Tech. rep. 1530. Computer Sciences Department, University of Wisconsin-Madison (cit. on p. 9).

Zhuo, Hankz Hankui, Qiang Yang, Derek Hao Hu, and Lei Li (2010). "Learning complex action models with quantifiers and logical implications." In: *Artificial Intelligence* 174.18, pp. 1540–1569 (cit. on p. 29).

Zöller, Marc-André and Marco F Huber (2021). "Benchmark and survey of automated machine learning frameworks." In: *Journal of artificial intelligence research* 70, pp. 409–472 (cit. on p. 6).

{2.7182818284}ϑφερτυθιοπσδφγηξκλ