Learning Visual Concepts with Fewer Human Annotations

Sébastien Ehrhardt

New College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Supervised by Professor Andrea Vedaldi

Michaelmas 2020

I dedicate this thesis to my uncle Jean-Claude Tavernier.

Acknowledgements

I am infinitely grateful to my supervisor Professor Andrea Vedaldi, who made this thesis possible. His guidance was extremely precious during this DPhil and I learnt a lot from his feedback. I cannot be thankful enough for the great support he provided me with throughout this journey and I feel honoured to have been his student.

I also thank my collaborators Aron Monszpart, Sylvestre-Alvise Rebuffi, Olivia Wiles, Kai Han, Oliver Groth, Niloy Mitra, Ingmar Posner and Andrew Zisserman. I greatly enjoyed working with each of them and benefited a lot from their experience. The different point of views they each brought to our work always helped it to further improve in quality. I particularly thank Olivia Wiles whose friendship kept me afloat in tough times.

In addition to collaborators, I thank all the VGG lab members for creating a nice working atmosphere where sharing ideas and helping one another is valued. I grew a lot working in this environment. I am especially grateful to Erika Lu, Arsha Nagrani and Sylvestre-Alvise Rebuffi for selflessly offering me their accommodation.

No amount of work would have been possible without the precious love of Shivani who helped me find the resources to always push myself more. She was the best shelter to forget about the worries of my DPhil and never failed to bring me support when it was needed. I also thank my family, my friends, Khushi and Zohan for their unconditional support and kindness.

I am very grateful to my British family in Blandford avenue. Living with them all these years always felt like being at home.

Finally I would like to thank ERC grant IDIU-638009 for financially supporting this DPhil.

Statement of Originality

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Sébastien Ehrhardt, New College

Abstract

This thesis explores the use of modern deep neural networks to learn visual concepts with fewer human annotations on data. While data is abundant and increasingly easier to collect, most deep learning methods need extensive human labelling to be trained, which is often costly and may require expert-level knowledge. In this thesis we explore alternatives to human labelling by considering synthetic data, as well as partially and completely unlabelled data. We will study these alternatives within two visual concepts related to human-level intelligence: *intuitive physics* and *object recognition*. For the former, we will focus on synthetic and unlabelled real sequences while for the latter we will focus on collections of images of natural categories that are scarcely annotated.

The first part of this thesis explores the ability of recent convolutional neural networks to learn physics through long term prediction. We first show that from limited visual data it is possible to make accurate future predictions of systems obeying the Newton's laws of physics. We propose to work on more complex synthetic and real data than standard benchmarks. For real data, we also develop an automatic self-supervised labelling method. Second, we propose a meta-learning pipeline that is able to infer environment properties from past experiences without any need for annotations. In particular, this understands causal concepts such as solidity. To conclude our work on physics we explore physical plausibility. Inspired from recent research on neural physics, we propose a generative architecture with a structured latent space that works on unlabelled data. Our model is not only quantitatively superior to prior art but has also a deeper understanding of the relations between objects in a scene.

The second part of this thesis is dedicated to object recognition. We consider the realistic scenario where data annotation is incomplete *i.e* lots of images are unlabelled. We make contributions in three different scenarios with decreasing levels of annotations. For each of them, we strongly advocate a systematic use of *self-supervision* as a pre-training step as well as *self-training* in the last step of training. We first introduce a novel semi-supervised learning method which performs optimisation alternatively on labelled and unlabelled data. Albeit simple, our method can work with as few as ten labelled instances per class. Next, we develop a clustering method that is initialised from supervision on other datasets. Clustering is then performed thanks to a new way of extracting pairwise similarity between two images: by comparing the rank of the most activated neurons of their network embedding. Finally, we extend the use of pairwise similarity to the nearest neighbours in the network embedding space and propose a clustering method that works *without* any supervised signal on different types of data.

Contents

List of Figures

Abbreviations

CNN ..... Convolutional Neural Network

GAN ..... Generative Adversarial Network

GMM ..... Gaussian Mixture of Models

GN ..... Graph Network

HoG ..... Histogram of Gradients

KL ..... Kullback-Leibler

LSTM ..... Long Short-Term Memory

MCMC ..... Markov Chain Monte Carlo

MLP ..... Multi Layer Perceptron

PCA ..... Principal Component Analysis

SSL ..... Semi-Supervised Learning

SVD ..... Singular Value Decomposition

VAE ..... Variational Autoencoder

VIN ..... Virtual Interaction Network

VMC ..... Visuomotor Control

VQA ..... Visual Question Answering

Chapter 1

Introduction

Vision is probably one of the sensors humans use the most. From an early age we are able to learn and make predictions using vision alone often without needing any external supervision. This enables humans and other intelligent beings to perform daily tasks such as object manipulation, dynamic forecasting, instance segmentation; where most *state-of-the-art* intelligent algorithms still struggle. Understanding perceptual vision would therefore totally enhance the performances of computer vision algorithms in a wide range of applications. A major step in that direction has recently been achieved with the development of convolutional neural networks (CNNs) (Fukushima, 1980; LeCun et al., 1989) and their successful application to large scale classification (Krizhevsky et al., 2012). This has enabled researchers to reach unprecedented performances on tasks including but not limited to image classification (Krizhevsky et al., 2012; He et al., 2016) or object detection (Szegedy et al., 2013; He et al., 2017). Hence, deep learning and particularly CNNs have become the golden standard to all computer vision tasks, with often startling performances, sometimes even outperforming expert humans (He et al., 2015). Thanks to CNNs, solving perceptual vision has never seemed so reachable. Yet, they face a major limitation. Most ground-breaking results using deep learning came at the cost of extensive image annotation. In fact, since they are initialised randomly, CNNs need some additional knowledge on each image, like a supervised signal coming from a label, to be able to make correct predictions.

Annotating data often comes at the expense of time, money and expert-knowledge which makes the application of supervised deep learning less scalable to real world problems. But labelling is not everything. While one can think of annotating each data sample with as much information as possible, it is also possible to incorporate other sources of information in the training. For instance, one could leverage some priors over the data distribution (Gidaris et al., 2018) or use human perceptual ro-

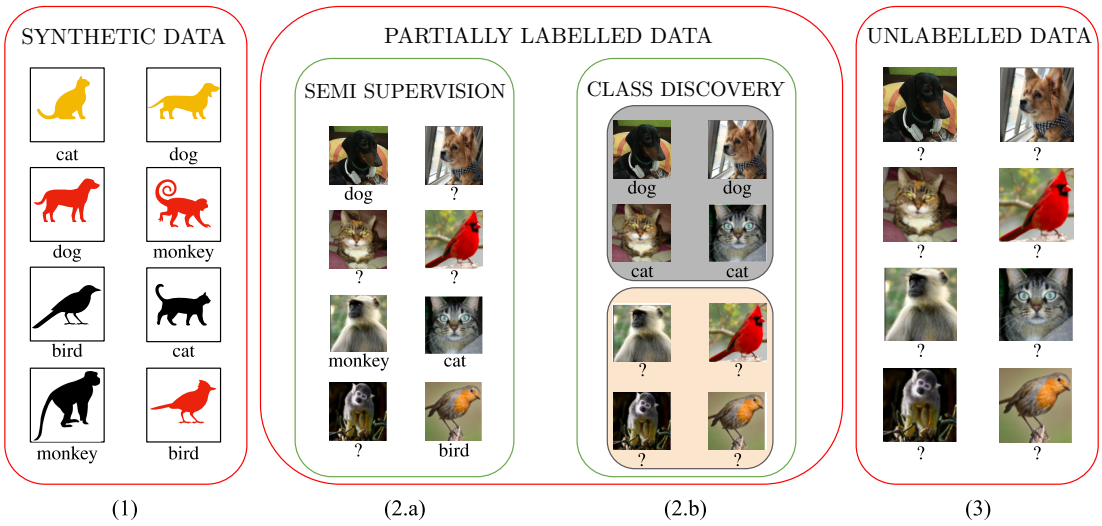


Figure 1.1: **Dataset scenarios we consider in this thesis.** From left to right: (1) **Synthetic dataset** which automatically provides annotation for each sample (chapter 3). (2) **Partially labelled set**. Two types are studied in this work; (2.a) **Semi-supervised** where for each category a few labelled instances are available (chapter 7). (2.b) **Class discovery**, where the dataset is split into a labelled part and an unlabelled part which categories are disjoint (chapter 8). (3) Completely **Unlabelled set** (chapters 4 to 6 and 9).

bustness under small deformations (Thewlis et al., 2017) in order to learn meaningful representations. If supervision is still required, human knowledge can come in handy to simulate (Shrivastava et al., 2017; Groth et al., 2018) and compensate the lack of available real data.

In this thesis, we will focus on deep learning applications to scenarios where annotations of images in a collection are incomplete or can be accessed from simulations. In particular, we consider three different types of image collections summarised in figure 1.1: (i) simulated data with complete ground truth, (ii) partially labelled data and (iii) unlabelled data. The main objective of this work is to make deep learning more scalable and automated by reducing the need of unautomated human annotations on each data sample. This goal is achieved by digging as much information as possible from the data itself using self-supervision or by generating simulated data.

This thesis will focus on two central concepts which are often easily grasped by humans: *intuitive physics* and *object recognition*. In what follow, we will introduce each concept within a separate section. For both, we will first present the motivations

behind our work before introducing each individual contribution of this thesis with its relevant chapter.

1.1 Intuitive Physics

Intuitive physics is defined by Kubricht et al. (2017) as: 'the knowledge underlying the human ability to understand the physical environment and interact with objects and substances that undergo dynamic state changes, making at least approximate predictions about how observed events will unfold'. In short it is the learning process behind humans perception of events and scenes that are controlled by the laws of physics. Recent works in cognitive science (Hamrick et al., 2016), demonstrated that perceptual information coming mostly from vision helped humans to understand the physics behind a scene, allowing easier manipulation of surrounding objects. While this understanding does not always fit with an accurate reality (McCloskey et al., 1983; Kubricht et al., 2017) it is often enough to perform daily activities. In particular, this ability can be very useful to agents that are tasked to perform complex interactions with their environment. Hence, intuitive physics is now a very actively researched domain as it could lead to more robust agents with human-like abilities. It is also often claimed that building machines that understand intuitive physics is a step towards human-level intelligence (Le et al., 2012; Lake et al., 2015). From now, the application of deep learning to intuitive physics will be referred to as *neural physics* for simplicity.

One particular advantage when it comes to physics is that the equations governing the physical world are well-known. Therefore, it is easy to run simulations through a physics engine with a graphic renderer such as Blender. Since Battaglia et al. (2013) most efforts in neural physics have been devoted to simulated data. This was in general enough to understand concepts such as stability (Lerer et al., 2016; Groth et al., 2018) or counterfactual reasoning (Baradel et al., 2020; Yi et al., 2020) which are often understood by infants (Hespos & VanMarle, 2012). However, less research has been dedicated to long term forecasting from visual inputs. Forecasting is important when it comes to decision making since it enables anticipation. Another bottleneck of past works in neural physics is its relatively rare application to real data. While simulated scenarios can provide realistic-looking data, they do not model the stochasticity of the real world. When real data were used (Wu et al., 2015, 2016a), it often came with prior knowledge over the scene and its components. This limitation is

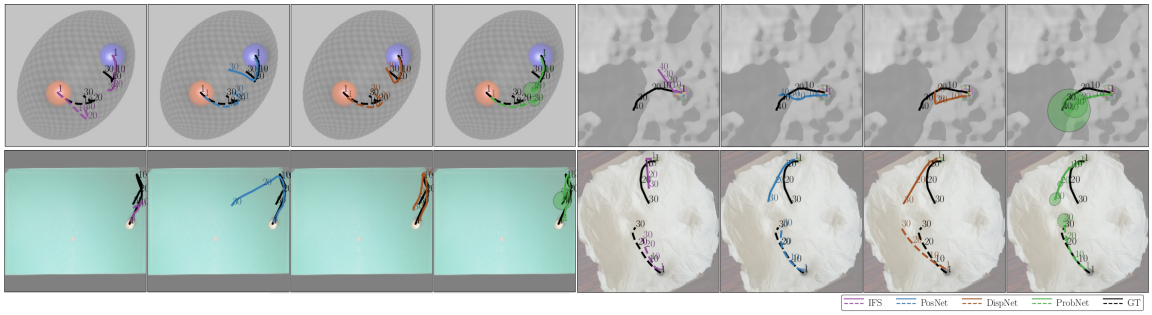


Figure 1.2: **Chapters 3 and 4: long term physical prediction.** Long term prediction is an inherently hard problem. Multiple predictions can be made depending on the complexity of the background. Each different colour indicates the prediction of a different method described in chapters 3 and 4.

particularly burdening since it requires to build a hand-crafted model of each scenario before running a neural physics engine.

In our work on intuitive physics, we will use long term forecasting at the core of our methods to learn physical concepts. In particular we try to address the limitations of past works in neural physics by answering the following questions: (i) can we learn long term predictors that work on synthetic and real data from visual inputs (ii) is it possible to learn causal concepts such as whether a block is a solid object or not from past experiences and (iii) can we generate a scene component-wise by enforcing physical plausibility and make realistic future predictions.

1.1.1 Long term extrapolation

While the equations governing the physical world are well understood, making long term predictions remains a challenging problem because of the chaotic nature of objects interactions. Indeed, non-trivial interactions such as collision or friction modify the trajectories of objects and mis-estimation can lead to important divergence between prediction and ground truth. While one could seek to first estimate all physical parameters of the components of a scene and use them to perform simulations (Wu et al., 2015, 2016a), it presupposes the knowledge of the existence of these components. Besides, as previously stated, small errors in estimation of any scene component's parameter also lead to important errors in future predictions (Yu et al., 2016).

In chapters 3 and 4 we propose a deep neural network architecture that makes forecasting predictions of physical properties such as position or angular velocity from visual inputs. We first propose to work on more complex scenarios than the ones in

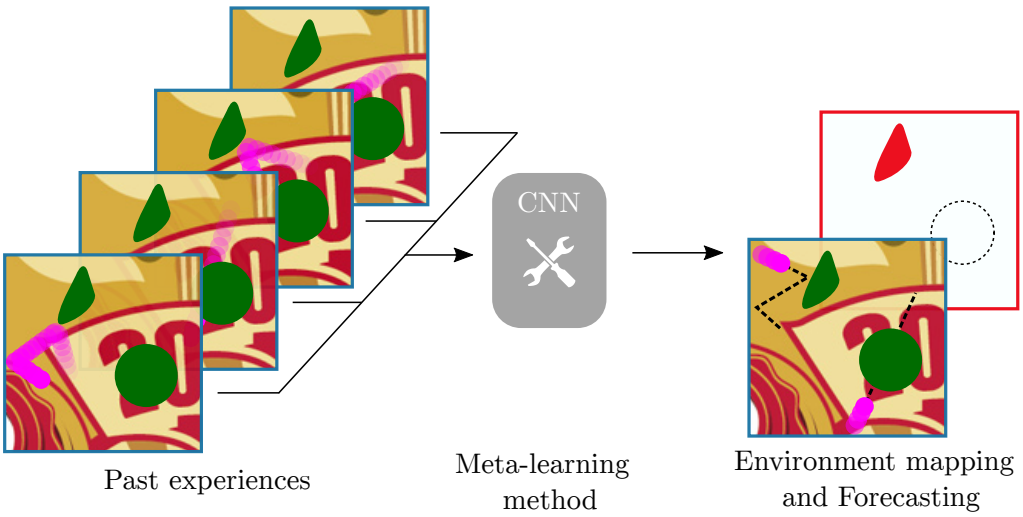


Figure 1.3: **Chapter 5: Unsupervised learning from past experiences.** Similar to natural intelligence, we task our system to infer physical properties of the environment from past seen experiences.

the literature. We collect both synthetic and real data of one to three balls rolling on different surfaces with complex interactions. Then, we empirically show that the use of a distributed tensor embedding is best to estimate motion uncertainty. As the complexity of the data increases our system qualitatively and quantitatively outperforms all other *vision-based* methods while being at par with methods that use ground truth objects and environment parameters. Besides, experimentally we find that regressing more parameters linked by mechanical energy is benefiting all algorithms.

Chapter 4 also proposes a method to extract objects' position labels reliably with a self-supervised learning method that uses equivariance and entropy minimisation. In particular, among other trackers, ours performs the best for all types of data with one or two objects. This tracker allows for the successful end-to-end training of our model without requiring any annotation on data. Finally the real data collected as well as the ground truth annotations on the test set are released online to help future researchers to investigate their own predictor.

1.1.2 Object properties from past experiences

To make long term predictions one also needs to accurately model the surrounding environment. While humans can easily abstract environment properties from visual observations, this is in general not the case for artificial intelligences. For instance,

to verify if an object is solid one could touch it or throw something at it to check its consistency. Then, inference can be made from the results of such experiment (*e.g* did it bounce ?). In particular, only the outcome of the experiment matters and no real supervision is needed in the process. While this problem seems trivial, it tackles a central concept of natural intelligence: *causality*. When observing a bounce one deduces that the cause of the bounce is the object itself, otherwise the system would have followed a free fall motion.

Chapter 5 takes a closer look at such problem and tries to build a system that is able to learn physical properties 'on-the-fly' from past experiences. In addition, we would like our system to obey three key properties. First, following the main thread of this thesis we would like to make long term predictions, hence building a system that is *temporally* invariant. Then, we would like our system to scale to larger images making it *spatially* invariant. Last, we would like our model to generalise to arbitrary number of objects, making the predictions *local*. Much like natural intelligences we want our full pipeline to be completely unsupervised. In practice, we build a meta-learning model which learns an abstract model of the environment from past experiences. To avoid using burdening recurrent architectures we compress past experiences videos into dynamic images (Bilen et al., 2017). Then, we use a meta-learning module that maps the experiences into an environment feature which helps making long term predictions. Finally, we design our pipeline such that we keep a distributed and interpretable representation which obeys the concepts mentioned earlier. More precisely, we use an internal tensor representation to model the physics of the system. In the end, the scene is rendered in two parts: a static and a dynamic part. This chapter contributes to work in unsupervised neural physics and causality by: (i) formulating key properties for physical predictors, (ii) proposing an efficient way to represent past experiences and use them to build an abstract state of the world. (iii) proposing a meta-learning architecture that is able to deduce environment properties from past experiences.

1.1.3 Physical Plausibility

A final concept related to physics that we approach in this first part is plausibility. Since we would like to approach this concept *without* annotations we cannot explicitly check the plausibility of a scene by verifying if it is violating the equations of physics. Instead we cast this as a perceptual problem. Traditional Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) or Variational Autoencoders (VAE) (Kingma & Welling, 2013) setups do not naturally provide an interpretable

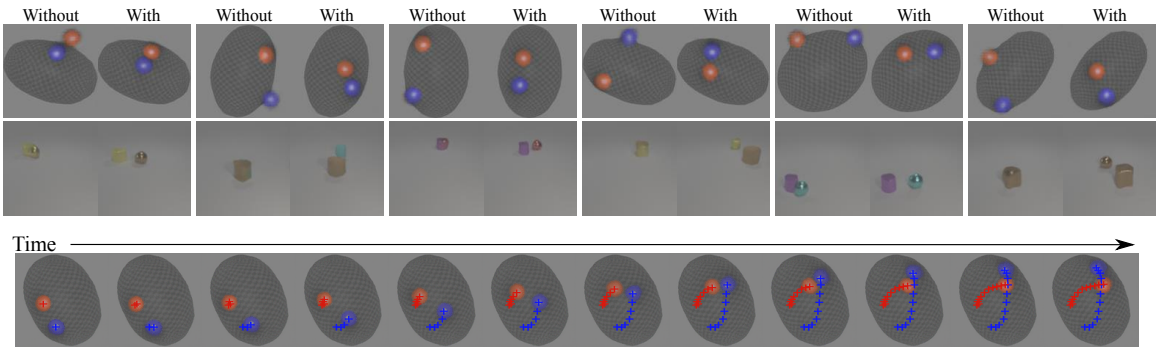


Figure 1.4: **Chapter 6: generating physically plausible scenes.** First two rows show samples with and without using our proposed correction module on static scenes and the third row shows a video sample generated from our method. Our method can model physic-related interactions between objects and their environment (first row and third row) or between two objects (second row).

latent space. Therefore, they only focus on realistic outputs. But it is also possible to build them so that they generate a scene component-wise, one object at a time. With the notion of an object then comes physics. To generate a scene one object at a time one needs to understand elementary physics. For instance, two objects cannot coincide at the same position in the real-world. They also lie on a surface as they cannot float in the air. Interactions are also perceptible in videos where objects' velocities are modified because of their interaction with other objects or with the environment. These concepts are further illustrated in figure 1.4.

In chapter 6 we propose a novel method to generate high-fidelity physically plausible scenes which would have such understanding. In this chapter, we propose a generative model equipped with a structured latent space. Physical plausibility is checked by a discriminator which is tasked to determine if a scene is real or fake. Our generator is also augmented with a module that models correlations between objects and between objects and the environment as well as a position loss. The latter learns the relation between an image and the location of individual objects in it. This forces each object to appear in the generated image. Both help to build a generative model which estimates physical plausibility without any need for annotations and provides an interpretable latent space suited for image editing. We show that our system outperforms all other object-centric generative methods and also standard GANs. In addition, our model naturally extends to the generation of long term videos of simulated and real world scenarios.

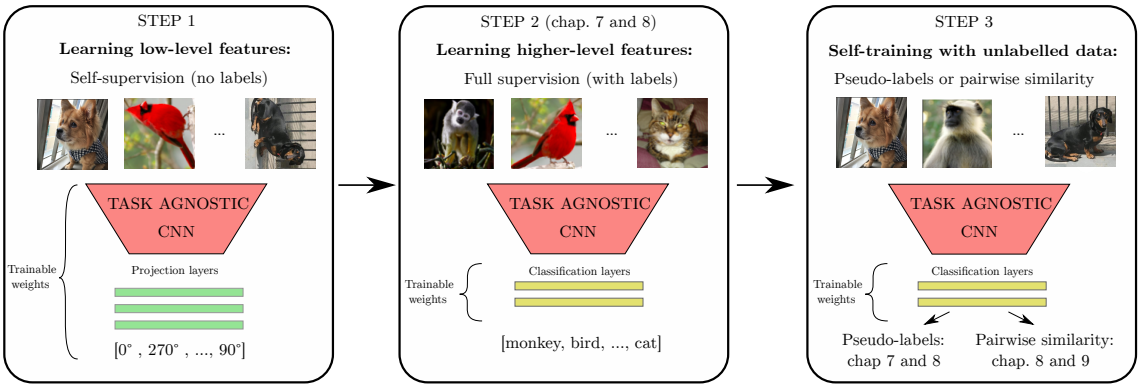


Figure 1.5: **Chapters 7 to 9: General pipeline proposed in this thesis to perform object recognition with fewer annotations.** In all our work, we leverage unlabelled data at the first and last step of our pipeline with an optional intermediate fine-tuning step on labelled data. More precisely, the first step uses self-supervision, *e.g* we solve a proxy task like predicting image rotation (Gidaris et al., 2018) to learn an image embedding. In the final step, we use self-training with pseudo-labels or pairwise similarity to produce a training signal for classification.

1.2 Object Recognition

Continuing the process of understanding what an 'object' is and what are its properties we pursue our work to object recognition. In this thesis, we restrict the scope of object recognition to image classification, a task that requires to determine which object class an image belongs to. From chapters 3 to 6, we limit our work to objects of similar visual appearance. However in the real world most objects have different visual appearances. If object recognition has been a central task in the development of deep learning (LeCun et al., 1989; Krizhevsky et al., 2012), it is also of primary importance in real world applications. Applications using object recognition include autonomous driving, fraud detection, face recognition, etc. In fact, since object recognition is a classification problem, its potential applications are not limited to computer vision. While classification using deep learning now sometimes surpasses human-level accuracy (He et al., 2015), these groundbreaking results are largely due to supervision using a label on each data. We argue that this is a limitation in practice. Since some datasets now contain millions or billions of images (Deng et al., 2009; Mahajan et al., 2018), labelling is becoming manually infeasible.

In a second part of this thesis, we will study visual object recognition on more

realistic scenarios than full supervision. Our main setting is a dataset featuring distinct objects or categories whose number is known *a-priori* but where labelling is incomplete. We also consider labels to be *rare* for some or all categories. This is of particular interest for scenarios where labelling is expensive; for instance when it requires expert-knowledge. For each work we find that self-supervision, an automatic way to learn an image embedding, *i.e* a lower dimensional representation of the data, is useful to initialise our network. In fact, the most recent advances in self-supervision (Gidaris et al., 2018; Chen et al., 2020a) have shown that the network embedding obtained from self-supervised pre-training is reaching supervised-level accuracy. Hence this thesis advocates an automatic use of self-supervision in the specific regime where labels are scarce. The use of self-supervised pre-training has two specific advantages: first it avoids overfitting and local minima by restricting the training to fewer network's layers and second it increases the speed of the training by back-propagating gradient through a smaller portion of the network. Another principle we use in this thesis is self-training or pseudo-labelling which can be seen as another form of self-supervision. More precisely, we use pseudo-labels extracted from a learnt predictor to hallucinate annotations on unlabelled data. We use two types of pseudo-labels: multi-class pseudo-labels in chapters 7 and 8 and pairwise similarity pseudo-labels in chapters 8 and 9. The first is used to attribute a category label to an image while the later predicts if two images are similar *i.e* if they belong to the same category. The general pipeline that we propose in this thesis can be found in figure 1.5. Our work in object recognition targets scenarios with decreasing level of annotations: semi-supervised learning, category discovery and clustering.

1.2.1 Semi-supervised learning

Chapter 7 introduces a two-step pipeline for semi-supervised learning algorithms in the case where there are very few annotations. While semi-supervised learning is a well-studied problem its applications to such a case remain rare and most algorithms in the literature exhibit degenerate behaviours. As noted in section 1.2 we make use of self-supervised pre-training in the first step to initialise our network. We then propose to alternate optimisation on the labelled and unlabelled set to increasingly improve the performance of our model. During optimisation we first train the classification layer on the labelled set with full supervision. Then we use pseudo-labels on the unlabelled set along with a consistency loss to train a larger portion of the network. We show that this pipeline can work on datasets with as few as ten labelled instances per class. Our algorithm standalone obtained results at par with the best methods

and *state-of-the-art* when combined to other SSL algorithms. Most notably, it works increasingly better as the initialisation gets better in quality and is robust to different types of architectures.

1.2.2 Category discovery

While semi-supervised learning has almost always existed alongside classification, a more recent task is category discovery. This task consists of using a labelled set to learn 'how to classify' on an unlabelled set. In practice, this sums up as a transfer learning task where the knowledge acquired by a neural network on a labelled set is leveraged to perform a clustering task on an unlabelled set. This is new since most works in transfer learning were mostly focused on transferring low-level embeddings, *i.e* coarse feature detectors, but rarely addressed learning a semantic embedding that is suited for transfer to unsupervised classification on other sets.

In chapter 8 we combine an efficient pipeline with a novel way to compare features extracted from a CNN to transfer between classification tasks. Inspired from the general pipeline presented in figure 1.5, we propose a three-step pipeline which starts from self-supervision on the combined labelled and unlabelled set. Then we perform supervised training on the labelled set. The last step uses a joint training on labelled and unlabelled data with a different classification head for each set. This time, we use pairwise similarity pseudo-labels to train our model on the unlabelled data while we train with full supervision on the labelled one. To generate pairwise similarity labels, we propose a new way to compare the activations from the penultimate layer of a CNN. Instead of computing the distance between the two embeddings, we show that it is possible to determine their similarity based on the order of their activations. Thus two embeddings will be considered similar if their k biggest activations are located at the same positions and vice versa. Finally by using pseudo-labels obtained from the clustering method, we show that we can use one network to classify all sorts of images (coming from the labelled or the unlabelled set) without prior knowledge of the origin of the image.

1.2.3 Clustering

Finally chapter 9 is dedicated to classification with no label at all: *clustering*. This task is particularly challenging since without any label, obtaining a training signal is a very ill-defined problem. Here, we use our experience from chapter 8 to perform clustering by identifying similarity between images. Starting from a data embedding

obtained from self-supervision, we classify samples by using similarity pseudo-labels obtained from the penultimate layer of a CNN. This time however, we define similarity as the closest neighbours of a sample within a batch. This work shows that computing similarity from the penultimate layer of a CNN is experimentally more robust than prior works where similarity was established from the prediction layer. Empirically we show that our method substantially benefits from a data augmentation technique which combines images as patches. Again this chapter shows how self-supervision combined with pairwise similarity pseudo-labelling pushes the limits of existing work. In particular, we show how pairwise similarity with closest neighbours can be used to learn how to cluster data *without* any need to annotate data. Applying our method to text classification, we also show how object recognition methods can impact classification in other domains. This further enhances the claim made in section 1.2.

1.3 Publications

Each individual chapter of this thesis introduces a publication listed as below (* denotes equal contributions):

Chapter 3: Taking visual motion prediction to new heightfields.
Sébastien Ehrhardt, Aron Monszpart, Niloy Mitra and Andrea Vedaldi. *In Computer Vision and Image Understanding Journal (CVIU)*. 2019.

Chapter 4: Unsupervised intuitive physics from visual observations.
Sébastien Ehrhardt*, Aron Monszpart*, Niloy Mitra and Andrea Vedaldi. *In Asian Conference of Computer Vision (ACCV)*. 2018.

Chapter 5: Unsupervised Intuitive Physics from Past Experiences.
Sébastien Ehrhardt, Aron Monszpart, Niloy Mitra and Andrea Vedaldi. *Technical report*. 2019.

Chapter 6: RELATE: Physically Plausible Multi-Object Scene Synthesis Using Structured Latent Spaces.
Sébastien Ehrhardt*, Oliver Groth*, Aron Monszpart, Martin Englecke, Ingmar Posner, Niloy Mitra and Andrea Vedaldi. *In Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

Chapter 7: Semi-supervised learning with scarce annotations.
Sylvestre-Alvise Rebuffi*, Sébastien Ehrhardt*, Kai Han*, Andrea Vedaldi and Andrew Zisserman. *In Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2020.

Chapter 8: Automatically discovering and learning new visual categories with ranking statistics.
Kai Han*, Sylvestre-Alvise Rebuffi*, Sébastien Ehrhardt*, Andrea Vedaldi and Andrew Zisserman. *In International Conference on Learning Representations (ICLR)*. 2020.

Chapter 9: LSD-C: Linearly Separable Deep Clusters.
Sylvestre-Alvise Rebuffi*, Sébastien Ehrhardt*, Kai Han*, Andrea Vedaldi and Andrew Zisserman. *Technical report*. 2020.

Chapter 2

Literature review

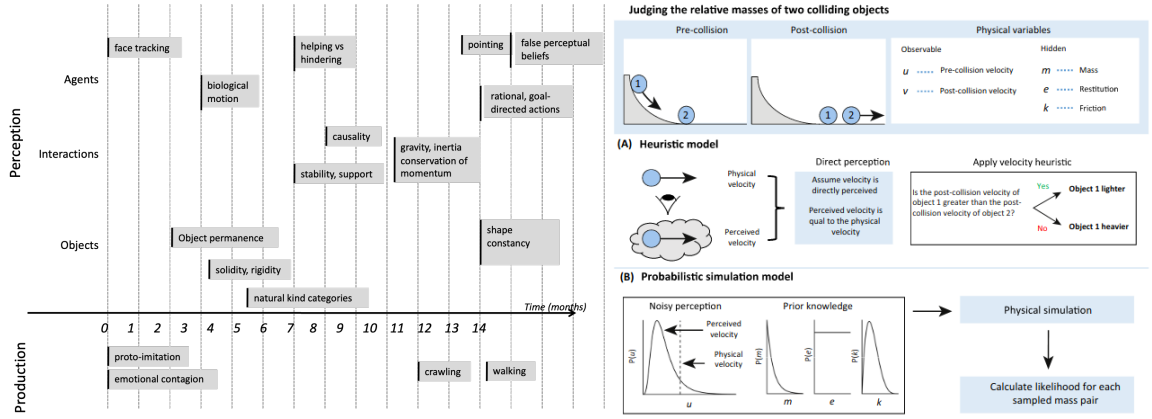
While this thesis is related to a broad range of computer vision applications, we narrow down our review of related works to the two dominant applications mentioned in the introduction with a greater focus on works that used deep learning. In this chapter we will split our study into two different sections following the main thread of this thesis.

In section 2.1 we will start by describing past approaches related to learning intuitive physics. Section 2.1.1 will review works in intuitive reasoning, in this subsection we will mostly consider the ones that approached 'qualitative' human reasoning with artificial intelligences. In section 2.1.2 we will take a more *in-depth* look into predictive models. In particular, we will see how machine learning has been used to learn models that can predict the dynamics of scenes obeying the laws of physics.

The second part of our literature review will be dedicated to methods in object recognition working with fewer labels. In sections 2.2.1 and 2.2.2 we will introduce the two main scenarios arising in the context of category recognition when dealing with limited annotations: semi-supervised learning and clustering. As this thesis has a strong emphasis on self-supervision we will then study in section 2.2.3 how self-supervised learning approaches were applied to the two aforementioned cases.

2.1 Learning physics from visual observations

From a relatively young age, infants develop an understanding of simple physical concepts such as *object persistence* (Baillargeon et al., 1985) or *gravity* (Kim & Spelke, 1999). Riochet et al. (2018) established a timeline of intuitive physics acquisition showed in figure 2.1(a). We see that most of these concepts are acquired from a very young age. Hence whether they are learnt or not is still actively researched (Kubricht et al., 2017). However, the probabilistic prediction model currently used in cognitive



(a) Riochet et al.

(b) Kubricht et al.

Figure 2.1: **Intuitive physics from cognitive science.** (a) **From (Riochet et al., 2018)**. Intuitive physics learning timeline: each box represents a study that shows intuitive physics acquisition in infants. (b) **From (Kubricht et al., 2017)**. The two current cognitive models for human physical predictions explained through a relative mass judgement example. (A) The heuristic model (not learnt) suggests that humans make predictions based on silent clues *e.g* the object that moves faster after the collision is the lighter. (B) The probabilistic inference model (learnt) which suggests that humans make stochastic predictions based on mental simulation and prior knowledge. This model is often preferred (Kubricht et al., 2017).

science (Hamrick, 2011; Battaglia et al., 2013; Sanborn et al., 2013)(see figure 2.1(b)) suggests that humans make predictions based on learnt priors and simulated physics. While the actual learning algorithm behind human understanding of physics is not yet agreed upon (Le et al., 2012), since the emergence of deep learning, neural networks seem to be the preferred method for the application of machine learning to intuitive physics.

2.1.1 Learning intuitive physics

As presented in figure 2.1(a), there are lots of branches related to intuitive physics. In the following we will divide works in the literature by their domain of application and their learning paradigm (active/passive). The default learning paradigm considered is passive, where the agent can make observations of the world without being able to interact with it. The works we will present primarily relate to figure 2.1(a). We will look at methods that trained machines to learn human-like abilities such as regressing physical quantities, inferring stability, learning from past experiences, deducing causality or inferring relations between objects.

Regressing physical properties from vision. Inferring physical properties of objects is central in natural intelligence to make predictions. Since humans can approach physical quantities such as mass (Hamrick et al., 2016) from their visual system alone, this suggests that it should be possible to learn physical quantites from visual inputs only. To tackle this issue, Bhat et al. (2002); Monszpart et al. (2016) used the equations of physics as well as the properties of objects to infer properties such as velocity or angular velocity from videos. Following the probabilistic mental simulation model, getting access to more physical quantities from vision was done through an internal physics representation (Wu et al., 2015). In Galileo, Wu et al. (2015) used a 3D reconstruction of the scene. After observing a video, the authors used Markov Chain Monte Carlo (MCMC) sampling of possible scenarios to pick the physical parameters (friction, mass) that best corresponded to the observation. While this method worked on real data, it also assumed knowledge of each component of the scene to be able to simulate. Hence it was not scalable. Other works (Brubaker et al., 2009; Pham et al., 2015; Zhu et al., 2016) showed that it is possible to estimate quantitatively the forces acting behind a scene by solving a constrained quadratic problem with a suitable model of the forces.

With the increasing popularity of deep learning, and its success in vision-related tasks (He et al., 2016, 2017), researchers incorporated CNNs into their method. Hence Physics101 (Wu et al., 2016b), the direct extension of Galileo (Wu et al., 2016a), replaced the 3D simulations with a physics engine by direct supervision over physical laws and quantities. CNNs were also used to estimate mass (Standley et al., 2017) or forces (Mottaghi et al., 2016a) from raw images much like human can do. Some related approaches also incorporated priors in the model to learn interpretable embedding. For instance, Wu et al. (2017) and Ye et al. (2018) either used direct regression of physical quantities or a physics engine to learn physical quantities through forward propagation and re-rendering. However even if the scene is well understood, estimating all properties and running a simulation through a deterministic model can lead to dramatic errors as evidenced by the conclusion of the work of Yu et al. (2016) on planar pushing datasets.

Predicting stability or support. Formally, stability is defined as the state of a physical system where its potential energy is at a local minimum. Intuitively this corresponds to a system which is unlikely to move or break under small constraints. Reasoning about stability or support also contributes to scene understanding. For instance, Gupta et al. (2010); Silberman et al. (2012) used support to build a graph

representation of a scene and understand the relationship between objects. Jia et al. (2013); Zheng et al. (2015) used stability reasoning to segment a scene and merge 3D segments together. Their assumption came from the fact that the world surrounding us is mostly stable. A plausible segmentation of the scene is therefore stable under the gravity constraint.

It did not take long until deep learning showed superior performance to human judgment for stability prediction (Zhang et al., 2016). Zhang et al. (2016)'s work however focused on predicting stability without studying the cause behind the predictions. The first work to explicitly reason about visual stability of tower of blocks with CNNs was done by Lerer et al. (2016). In particular after simulating data to supervise their system, they used a sliding gaussian blur window to visualise the instability areas as inferred by their model. Deep learning applied to visual stability and stacking was later applied to real data and robotic manipulation (Li et al., 2017) and support (Groth et al., 2018).

Active learning from experience. While in this work we suppose that our agents are learning *passively* from observations of the world, it is also possible to learn physics through interactions. For instance, if it is possible to interact with the environment either directly or through simulations, then it should be possible to perform experiments much like humans would do. This could be done for instance to determine physical properties of an object such as mass or rigidity.

Agrawal et al. (2016) was among the first approaches to propose using robots and active manipulation to learn the concept of poking. Their approach was particularly interesting as they work on unlabelled real footages. In their paper, they tasked the robot to reach an end image from a starting point by performing pokes only. This was done by first collecting videos of random pokes and then learning an inverse dynamic model. Through manipulation, the robot then learnt concepts such as solidity, rigidity and pose. Similarly, Pinto et al. (2016) learnt active manipulation through poking, grasping or pushing. However their method was only designed to verify if low-level features learnt from active manipulation could be transferred to image classification tasks. They showed that their self-supervised method achieved similar or better results than passive learning on supervised sets.

Denil et al. (2017) used deep reinforcement learning to perform experiments that would help to answer visual questions such as 'which is heavier?'. It was completed by PHYRE, (Bakhtin et al., 2019) who proposed a new dataset for a more extensive

evaluation of reinforcement learning agents in intuitive physics. The dataset comprises static and moving objects and the agent is tasked to touch dynamic objects so as to reach a given goal. This emulated the concept of *goal-directed* actions (see figure 2.1(a)) learnt by infants around one year old. A potential application of active learning of intuitive physics from vision is Visuomotor Control (VMC). In such a scenario, the agent can interact with the world but receives inputs from vision only. For instance, VMC conditioned with an end-goal image (Groth et al., 2020) learnt concepts such as stacking and grasping from raw images alone. Interestingly, this work posterior to ours also proposed to use dynamic images (Bilen et al., 2017) to summarise experiences.

Causality, counter-factual physics, and object permanence. Learning the causes behind a motion or the presence of certain objects is also fundamental in human intelligence. The theory of causality has been well established both from a human (Pearl, 2009) and a machine learning (Schölkopf et al., 2012; Lopez-Paz et al., 2015; Mooij et al., 2016) perspective but has generally been applied to non-physical scenes (Lopez-Paz et al., 2017). Causality in neural physics is often approached via *counter-factual* reasoning. Through *if* statements, counter-factual physics checks whether a system has acquired an understanding of causality by predicting the realisation or the disappearance of an event. Mottaghi et al. (2016b) used it to predict the resulting movement from the application of a force onto an object appearing in a static image. CLEVRER (Yi et al., 2020) and COPHY (Baradel et al., 2020) both proposed new datasets to learn counter-factual physics either from Visual Question Answering (VQA) or by predicting the results of an action of a force to a scene. These last methods were however limited as they assumed that the system was aware of all objects appearing in the image.

Another emerging trend in neural physics studies whether a video is in violation of expectations or not. This checks learning of concepts such as solidity or object permanence. For instance, IntPhys (Riochet et al., 2018) proposed a benchmark where the task was to predict whether a video was physically plausible or not. Piloto et al. (2018); Smith et al. (2019) stayed closer to the original definition by predicting whether the expectation over a video matches the realisation. CATER (Girdhar & Ramanan, 2020) also proposed a benchmark that checks for object persistence by predicting the final location of a snitch that can be hidden by other objects at the end of a video.

Objects relations. Finally, object relations is a field though less related to the above but tightly linked with causality. While recent works in unsupervised objects segmentation (Engelcke et al., 2019; Greff et al., 2019) managed to disentangle different components in an image using *analysis-by-synthesis*, they often fell short of explaining the relations between objects. Past approaches in relation reasoning were generally an application of VQA. While there have been a number of datasets proposed for static visual reasoning (Winograd, 1972; Andreas et al., 2016), CLEVR (Johnson et al., 2017) recently dominated studies in relation reasoning. This dataset comprises high definition static images of cluttered table tops and contains nearly a million of questions about scene understanding for about 100,000 scenes. In general, researchers used *attention* modules (Yang et al., 2016b; Hudson & Manning, 2018) or explicit spatial relation encoding (Santoro et al., 2017) to reason about objects. Reasoning can also be made from seeing videos of the world. Several approaches disentangled different entities in dynamic systems or videos and learnt about relationships between them in order to predict future motions (Kipf et al., 2018; van Steenkiste et al., 2018; Li et al., 2020), activites (Baradel et al., 2018) or to track objects (Kosiorek et al., 2018; Fuchs et al., 2019). More refined than objects one can also seek to decompose objects into physical parts with different materials or properties (Liu et al., 2018).

2.1.2 Predicting Dynamics

Recent models in cognitive science suggested that in certain circumstances humans make predictions from simulated experiences (Battaglia et al., 2013; Sanborn et al., 2013). This brought the idea that human would have developed an *intuitive physics engine* which is at the core of physical reasoning. Hence building machines that can accurately predict dynamics is a central problem in machine learning. In general, researchers operated either directly on physical quantities to learn an approximate physics engine or learnt an implicit embedding from vision in order to make predictions. Here we will first introduce works that aimed to replace physics engines with deep neural networks. Then we will get closer to the goal of this thesis: *learning from vision*. We will tackle successively supervised and unsupervised learning of dynamics from vision.

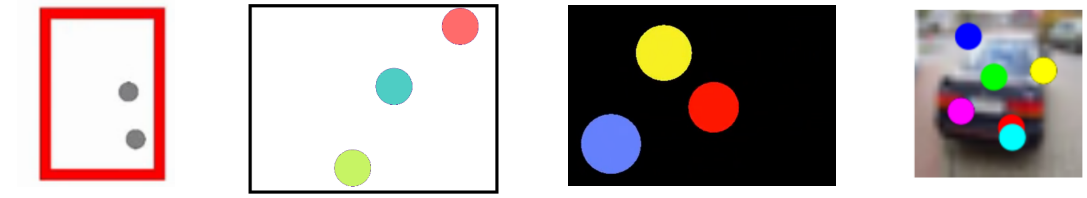
Neural networks as approximate physics engines. The first milestone in this area was achieved by Grzeszczuk et al. (1998) who quantitatively showed that solving numerical equations to simulate physics could be approached by using neural networks

to regress the next state of the world. Some later works (Battaglia et al., 2016; Chang et al., 2017b) also reused neural networks with good long term predictions. They mainly differ from prior art as they considered object relations as a form of graph networks (GNs) (Battaglia et al., 2018). In fact, both papers used two multi-layer-perceptrons (MLPs), one to model the interaction between two objects, and after summation of all possible interactions, the second MLP was tasked to translate the final interaction to a interpretable output *i.e* a translation. GNs were later reused in more complex 3D scenes (Mrowca et al., 2018), lately achieving very realistic results on fluid dynamics modelisation (Sanchez-Gonzalez et al., 2020). They were also successfully applied to learning the equations of physical systems and outperformed human-made models (Cranmer et al., 2020).

While GNs are now the dominant application to approximate a physics engine it is worth noting that some works also used alternative architectures. For instance, Rempe et al. (2020) used a 3D point cloud embedding to learn the dynamics of 3D objects; Tompson et al. (2017) used CNNs to solve the partial differential equations of motion to simulate fluid dynamics. Alternatively Ladicky et al. (2015) used random forests to approximate fluid dynamics. A new trend of neural physics also emerged extending training to Lagrangian (Lutter et al., 2018) and Hamiltonian (Greydanus et al., 2019) physics by constraining the training loss to fit the relevant equations of physics.

One bottleneck of these works however is that they did not work on perception but assumed all underlying physical quantities to be known. This is unlikely to be the case for the perceptual system of humans which by default takes raw visual inputs.

Learning perceptual dynamics with direct supervision. Learning from raw pixels alone is a much harder problem than using a refined signal such as the positions of the objects, their properties, etc. Indeed one needs to first associate every pixel in an image to individual objects, predict the properties of these objects and propagate this in time. While unrelated to physics, Oh et al. (2015) was a pioneer work in action-conditioned video prediction that managed to make accurate video predictions in Atari games and used the predictions to successfully train a Q-controller for playing. Slightly more related to physics, Finn et al. (2016a) learnt a conditional video predictor which learnt the results of physical actions (push, poke, grasp) of a robotic hand over an object. However these works did not consider physical quantities alone but were rather interested in accurate video prediction.



Fragkiadaki et al., (2016) Chang et al., (2017b) Battaglia et al., (2016) Watters et al., (2017)

Figure 2.2: **Sample scenarios from related work.** The methods we compare against in chapters 3 and 4 generally featured very simple visual scenarios. In this thesis we propose scenarios with higher visual complexity.

Fragkiadaki et al. (2016) proposed to learn a physical predictor from vision only. They used a CNN combined with two Long Short-Term Memory (LSTMs) (Gers & Schmidhuber, 2001) networks to predict the next position of individual objects. Although this work was based on visual inputs combined with additional forces, it still had strong priors over the scene. First, it assumed the object to be at the center of the frame which induced a good tracking of the object. Second, it re-rendered the scene at every time step so as to feed new inputs to the LSTMs. Hence this still assumed strong prior knowledge of the scene rendering.

Another significant advance was achieved by Watters et al. (2017). The Virtual Interaction Network (VIN) proposed in their paper used Battaglia et al. (2016)'s Interaction Network at the core of its system and managed to make accurate predictions of dynamics. In addition to accurate long term predictions, some work empirically showed (Zheng et al., 2018) that the learnt visual embedding of VIN was encoding for physical properties. Similar models can also be used by other systems for planning without requiring a physics engine (Kansky et al., 2017; Silver et al., 2017; Sanchez-Gonzalez et al., 2018). We however show in this thesis that VIN only works for simple synthetic environments. When dealing with more stochastic scenes this paradigm fails to learn a meaningful representation.

Unsupervised physics from vision. It is also possible to learn physics from unlabelled data by incorporating the right prior. Inspired from work using computer vision as inverse graphics (Kulkarni et al., 2015) some advances suggested representing *physics as inverse graphics* (Finn et al., 2016b; Janner et al., 2018; Wang et al., 2018; Kossen et al., 2019; Jaques et al., 2020). Finn et al. (2016b) proposed to learn a visuomotor from a set of learnt keypoints that represented parts of movable objects. Janner et al. (2018) and Jaques et al. (2020) both worked in very similar ways. They first extracted a vectorised representation for each individual object. This was

then propagated thanks to a neural physics engine whose architecture was similar to Battaglia et al. (2016); Chang et al. (2017b). The final step learnt a rendering function that mapped back to the image space so as to match future frames. Ye et al. (2019) also worked in similar ways as the two later works. However they used a pre-trained representation rather than a handcrafted one as a form of a supervised object detector. Such pre-trained representation decreased the level of supervision from the two aforementioned papers. Finally, another prior can be physics itself. For instance, Stewart & Ermon (2017) used the equations of physics to discover free-falling objects in videos. This work however is limited to a single free falling object or constant velocity ones. When learning an encoder Guen & Thome (2020) also showed that it was possible to learn an encoding that split the motion of objects in two parts. Namely an explainable part governed by known PDEs and an unexplainable part where stochasticity comes from.

In this thesis, we go beyond pre-existing works in chapters 3 and 4 by considering more challenging scenarios than prior art (see figure 2.2). We also differ from past research as we consider a tensor representation to model internal physics as opposed to a vectorised one (Fragkiadaki et al., 2016; Watters et al., 2017). In addition, chapter 4 is among the first works on intuitive physics to work on unlabelled real data. We are also among the first works to jointly learn physical quantities that are linked by mechanical energy and show this benefits all methods. We extend prior research on causality in chapter 5. This through a novel meta learning method that learns environment properties 'on-the-fly' in a completely unsupervised manner from passive observations. In particular, this differs from the most related work of Baradel et al. (2020) as we do not assume knowledge of the components of a scene but rather try to infer all properties from unlabelled sequences. Finally our work in chapter 6 is the first to use GNs to model object relations in a completely unsupervised way from a generative architecture. Other research generally fell short in doing so (Anciukevicius et al., 2020) or used supervision to learn what is physically plausible or not (Riochet et al., 2018).

2.2 Object recognition with fewer labels

From figure 2.1(a) we can see that natural kind categories is another concept understood by humans at a very early stage of their development. In computer vision, visual object recognition or image classification is perhaps one of the most studied

fields in computer vision. Since the ground breaking results of Krizhevsky et al. (2012), deep learning has been, if not the only one, at least the dominant method applied to object recognition. From then on, Imagenet (Deng et al., 2009) has remained one of the most used datasets in image classification. However this dataset featuring millions of images is extensively annotated. This probably took years to prepare which is not scalable to daily applications. Fortunately, from the early ages of classification, researchers have looked at ways to learn algorithms with data where labels were missing. This section will delve into more details of the methods that tried to do object recognition with fewer human annotations. We will study scenarios with decreasing levels of annotation from semi-supervision in section 2.2.1 to unsupervised clustering in section 2.2.2. Finally we will introduce in section 2.2.3 recent techniques that facilitate object recognition by learning an embedding on unlabelled data using self-supervision.

2.2.1 Semi-supervised learning

If for all categories a few images are still labelled, then the problem is called *semi-supervised learning* (SSL). In this context, one would generally use the labelled data as anchor points and leverage the unlabelled data to extend the representation to a more complete distribution. Chapelle et al. (2006) summarised all past major contributions in semi-supervised learning and described the main assumptions they used. In this subsection, we will divide works by how they relate to three of the assumptions presented in Chapelle et al. (2006): *low-density separation*, *graph* and *generative* methods.

Low-Density Separation. This first class of methods tries to push away the decision boundary from all, labelled *and* unlabelled, data points. One way to proceed is to use the fact that two data points close to each other should share the same classification (called the *continuity assumption* (Chapelle et al., 2006)). Therefore incorporating noise either in the data embedding or in the data itself should push the decision boundary off any data point. Rifai et al. (2011) proposed to use this assumption to regularise their contrastive auto-encoder to make it robust under small variations of the input. This was achieved through a penalty term on the loss Jacobian. Other works (Bachman et al., 2014; Rasmus et al., 2015) proposed to incorporate noise at the embedding level and then compared to 'clean' embeddings. The goal was to learn a classifier that would be invariant to small noises and would better represent the semantic of the images. For instance, Bachman et al. (2014) compared the output of

the unmodified network to multiple versions where some noise was randomly added at different layers. Rasmus et al. (2015) incorporated noise at every layers and used a denoising decoder to clean the intermediate representations and finally compared to the unmodified encoding.

Closer in mind to the continuity assumption at the data level, Sajjadi et al. (2016) proposed to compare the predictions of different augmentations of the images where various intermediate noises were also applied at the output of different layers. This work also used a mutual exclusivity loss, which concept was similar to entropy minimisation used in other work (Grandvalet & Bengio, 2005). This forced the prediction at every point to be as confident as possible, hence pushing the decision boundary away from any unlabelled point. Laine & Aila (2017); Miyato et al. (2019) also reused similar assumptions to perform SSL. In Laine & Aila (2017) the network was also trained to be invariant to data augmentation. In particular, the temporal model used, for each image, an exponential moving average over past predictions. The distance between the moving average and the new prediction of the network was then minimised. In addition, it introduced a consistency loss gradually by progressively augmenting its weight into the final loss. Miyato et al. (2019) extended the concept to adversarial perturbations (Szegedy et al., 2014; Goodfellow et al., 2015). Instead of choosing from a known set of transformations the authors picked the one that was maximising the network confusion (*i.e* the one that lead to the maximum change of the network output). Adversarial examples were also used to generate two views of the same input image as in co-training (Qiao et al., 2018). Following Laine & Aila (2017), Mean-Teacher (Tarvainen & Valpola, 2017) also proposed an ensembling technique. While the Temporal ensembling model (Laine & Aila, 2017) proposed to apply a weighted average over the model's predictions, Mean-Teacher suggested to apply it directly to the weights of the network. In this paper, the 'teacher' was an exponential moving average of the student model, the raw network. The task was then to reduce the prediction gap between the student and the teacher.

More recent trends in SSL showed that stronger augmentations can lead to state-of-the-art results on current SSL image benchmarks (Berthelot et al., 2019b,a) . As well as ICT (Verma et al., 2019), these methods used MixUp (Zhang et al., 2018) data augmentation which creates new samples from the interpolation between two images. The newly created samples are then assigned a label which corresponds to the same interpolation between the labels of the two images. Besides MixUp, Berthelot et al. (2019b,a) also used other recent augmentation techniques such as

AutoAugment (Cubuk et al., 2018) or RandAugment (Cubuk et al., 2020) as well as self-supervision such as rotation prediction.

Closely related to entropy minimisation, self-training or pseudo-labelling is also relatively popular in SSL. In general, it consists of first training a predictor on the labelled data and then assigning to unlabelled data the label of its most confident class as predicted by the predictor. Since hard 0-1 labelling is used, it therefore imposes to minimise entropy on unlabelled data. The first successful application of this technique to deep learning can be found in Lee (2003). Rosenberg et al. (2005) also showed that this technique can be extended to object detection. While Oliver et al. (2018) showed that pseudo-labelling underperformed compared to other techniques, when combined to stronger augmentations it actually lead to current state-ot-the-art results in SSL (Arazo et al., 2019; Sohn et al., 2020).

Graph and generative methods. Early works in SSL considered label-propagation from connected graphs (Blum & Chawla, 2001; Zhu et al., 2003). The idea behind is to assign to each data point the label of its closest sample in the labelled set. To do so, one would first build a graph connecting each data point. A weight representing the similarity of each pair of samples is then affected to each edge of the graph. Then an algorithm such as MinCut (Blum & Chawla, 2001) is tasked to find the best cut of the graph which minimises some energy function. Prior method using graphs and deep learning (Weston et al., 2012) proposed to learn a deep network which would be equivariant with respect to the graph metric. The embedding of the neural network would retain properties of the graph such as proximity or separation while simultaneously minimising a cross entropy loss on the labelled data. Iscen et al. (2019) proposed to generate pseudo-labels from the adjacency graph constructed from the network embedding. After first optimising on the labelled data they computed the adjacency graph from the network embedding. Then, they used this graph to generate pseudo-labels for the unlabelled data and iteratively refined it.

Another class of models tries to build a model of $p(x|y)$, the conditional distribution of data with respect to each class. These methods are referred to as *generative* methods. Kingma et al. (2014); Maaløe et al. (2016) used a conditional VAE to disentangle classes from style in images. CatGAN (Springenberg, 2016) used a GAN at the core of its system. More particularly, it modified the discriminator so as to be a multi-class classifier. Then the discriminator was tasked to be confident on real data by minimising entropy on these data while maximising entropy on the generated

samples. It was also helped in its task by minimising cross entropy on labelled examples. The generator however, was tasked to confuse the discriminator by producing samples which would result in low entropy as per discriminator predictions and would also be equally distributed among all classes. Similarly, Salimans et al. (2016) further improved CatGAN by augmenting the number of classes by one that corresponded to generated samples and additionnally used more GAN regularisations. The task assigned to the discriminator was then to correctly classify labelled data to their class, unlabelled data in any of the classes but the last one and generated samples in the last class.

Finally, methods that do not belong to any of the categories specified in Chapelle et al. (2006) include: autoencoder pre-training with sparse learnt codes and supervised finetuning (Ranzato et al., 2007), external memory to create class templates (Chen et al., 2018) and gradient alignment between unlabelled and labelled data (Jackson & Schulman, 2019).

2.2.2 Clustering

Identifying visual categories from unlabelled data has been a long standing issue in machine learning. While there is a multitude of well-known clustering algorithms such as *k-means* (MacQueen, 1967), Mean-Shift (Comaniciu & Meer, 1997), DBSCAN (Ester et al., 1996) or Gaussian Mixture of Models (GMM) (Dempster et al., 1976) all methods suffer from the *curse of dimensionality* (Beyer et al., 1999) when dealing with images. Indeed, most of these methods are primarily using the Eucledian norm which is meaningless for data such as images that are lying in spaces with at least thousands of dimensions. However, current deep learning methods remain heavily inspired from original works in clustering, most particularly *k-means* (MacQeen, 1967) or spectral clustering (Ng et al., 2001). In this subsection, we will first review the methods using centroids and pseudo-labelling, and the ones using pseudo-similarity. Then we will tackle more theoretical approaches using information theory criterions and finally introduce a new paradigm for clustering.

Centroid and pseudo-labelling based methods. Inspired from centroid based methods such as *k-means* (MacQueen, 1967) and parametric dimensionality reduction t-sne (Van der Maaten & Hinton, 2008), DEC (Xie et al., 2016) used a pre-trained embedding from an auto-encoder and used *k-means* to initialise centroids. Centroids were then used to create pseudo-labels to train a neural network with a cross entropy loss. Then, they jointly optimised the deep network's weights and the centroids.

DCN (Yang et al., 2017) proposed an approach closer in mind to *k-means*. Following DEC (Xie et al., 2016), DCN used a pre-trained auto-encoder. Then, it alternatively optimised the network's weights, the cluster assignments and the centroids. Network's weights were optimised to minimise the clustering objective as well as a reconstruction loss to avoid degenerate cases. This model was further improved by forcing the embeddings used for clustering to retain structural information thanks to a reconstruction loss (Guo et al., 2017). Similarly, DEPICT (Ghasedi Dizaji et al., 2017) also used a reconstruction loss as well as pseudo-labels. Note that this work did not use centroids but updated the cluster distribution based on the assumption that the data was uniformly distributed among clusters. However they still used *k-means* to first learn clusters and probability assignments and pre-trained with a reconstruction loss such as Guo et al. (2017). ADC (Haeusser et al., 2018) also learnt centroids. This time, they did not pre-train their network with an auto-encoder but used a consistency loss between images and their augmented version as a pre-training objective.

Finally, submitted concurrently to our work Van Gansbeke et al. (2020) proposed a method that used self-labelling after the clustering method. This work shares a lot of similarity to the work we will present in chapter 9, most notably in the use of the self-supervised pre-training method and the nearest neighbour similarity. However, they used a significantly different data augmentation and their loss also aimed to maximise entropy across all classes assuming the images were equally distributed among classes. Besides, the results they reported using the same self-supervised pre-training method as ours underperformed our proposed algorithm.

Pairwise similarity. The very first step of spectral clustering is to build an affinity graph of the data points before finding the optimal cut. Prior works generally used Spectral Vector Decomposition (SVD) to reduce the dimensionality of the Laplacian matrix of the graph. Tian et al. (2014) empirically showed that the embedded space computed from SVD could be learnt instead. From a stacked auto-encoder with sparsity constraints, they showed that it was possible to remove some nodes of the graph which eased clustering. Besides, the algorithm was shown to be linear with respect to the number of inputs as opposed to the quadratic growth of SVD. JULE (Yang et al., 2016a) and FINCH (Sarfraz et al., 2019) started from an over-clustering of the data with *k-means* and then iteratively learnt an embedding through agglomerative clustering. However these methods still required to build a similarity matrix of all the data points which grew quadratically in size with the number of points. Taking

advantage of the relative small size of mini-batches, DAC (Chang et al., 2017a) learnt to cluster and embed data at the same time. Starting from the assumption that images either belonged to one cluster or did not, it used pairwise similarity in the embedded space to perform clustering. In fact, for each sample in the mini-batch, the closest embedded samples were forced to have the same embedding and similarly the most dissimilar embedded samples were pushed away from each other. Then, the cluster assignment was computed as the location of the maximum activated neuron of the CNN output. Thanks to the small size of the mini-batches often used in deep learning, DAC can learn an embedding with relatively low memory footprint by only using subsamples of the dataset.

Information based criterion. Like the *expectation-maximization* model, one could seek to maximize the likelihood of the data with respect to the model's parameters. VaDE (Jiang et al., 2017) enforced a GMM on the embedding of a VAE so as to disentangle different classes. Other works tried to maximise mutual information (Cover & Thomas, 2006) as defined by Shannon entropy. Some approaches using neural networks (Bridle et al., 1992; Krause et al., 2010; Hu et al., 2017) maximised the mutual information between an input and its lower dimensional embedding. Another possible way is to consider the mutual information between pairs of similar inputs. IIC (Ji et al., 2019) successfully learnt a clustering algorithm from unlabelled images by generating artificial pairs with different transformations. One particular achievement of this work was that it worked *without* using any pre-training while previous methods relying on mutual information such as IMSAT (Hu et al., 2017) used a low-dimensional embedding sometimes obtained from supervision.

Discovering new categories. Clustering often starts from the assumption that no annotations are available on the data. But one could learn what clustering means in a supervised way before applying a clustering algorithm to a new set of data. *Novel category discovery* casts clustering as a transfer learning problem from supervised classification. In this context, one would be given two sets. One set would contain labelled data onto which one would run a fully supervised classification algorithm. The other set, which categories are generally disjoint from the first, is the set of interest where practitioners would seek to identify different categories.

KCL (Hsu et al., 2018) proposed a pipeline that used both metric learning and clustering objective optimisation. In this work, a similarity function was first learnt on the labelled set. Then, joint optimisation was performed: a standard cross entropy

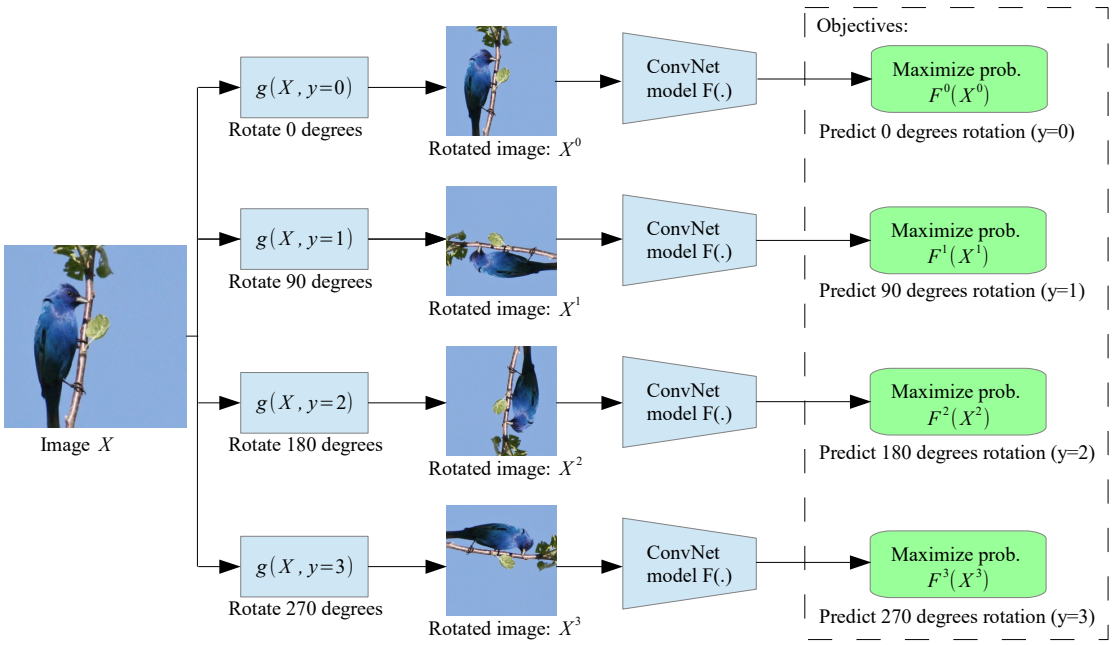


Figure 2.3: **From Gidaris et al. (2018): RotNet pipeline**. A random rotation chosen among {0°, 90°, 180°, 270°} is first applied to each input image (second and third column). The network is then tasked to predict which rotation was applied to the image (fourth and last column). After pre-training the last classification layer is replaced by one suited for the task.

loss was applied on the labelled data while the pairwise similarity was used on the unlabelled data to determine whether pairs of points belonged to the same category or not. Pairs of points were forced to have the same distribution as measured by Kullback-Leibler (KL) divergence if similar or different otherwise. This was further extended by Hsu et al. (2019) who assumed that the joint distribution of two points drawn independently was the product of the distribution of each point. Defining the similarity of two points X, Y as the marginal distribution over the classes $P_c(X, Y)$, they used binary cross entropy to minimise this similarity. Assumption of independence resulted in $P_c(X, Y) = P_c(X) \times P_c(Y)$ which forced low entropy predictions favouring clustering. Closer in mind to DEC, DTC (Han et al., 2019) first learnt how to do centroid clustering on the labelled set and after applied it to the unlabelled set. In addition to a clustering method DTC also proposed to learn the number of clusters in the unlabelled data by choosing the one that minimised a *cluster validity index*.

2.2.3 Self-supervision for category recognition

Self-supervision consists of learning a representation of the data from unlabelled samples. In general, one seeks to optimise a pretext task to learn a meaningful embedding. The range of proxy tasks is very large. For instance researchers have used patch context and jigsawpuzzles (Noroozi & Favaro, 2016), inpainting (Pathak et al., 2016), clustering (Caron et al., 2018), colorisation (Larsson et al., 2017), generation (Donahue & Simonyan, 2019), predicting transformations (Gidaris et al., 2018) or constrastive loss (Oord et al., 2018; Chen et al., 2020a). Self-supervised learning methods often learnt an embedding which is *linearly separable*. This means that from the representation extracted from the pre-trained network (generally at the penultimate layer), it is possible to learn a linear classifier that correctly classifies the different visual categories. In our work, we use RotNet (Gidaris et al., 2018)(see figure 2.3) as the self-supervised learning method. This method assumes that the images are generally shot upright. Therefore it should be easy to identify when an image of a given category has been rotated. From different rotations of an image, the proxy task of RotNet is then to predict the rotation that was applied. In this subsection we will be interested in past works that, like ours, used self-supervised pre-training to perform either SSL or clustering.

Application to SSL. One of the main issues with SSL and clustering is that because of lack of annotations, it is likely that the representation will collapse to degenerate solutions. For SSL in particular, since labelled data are available, the higher the number of weights in the network the more likely the overfitting on the labelled set. Self-supervised pre-training is generally missing in semi-supervised learning methods whereas SSL has now become a standard of evaluation for self-supervised methods. Since the work of Gidaris et al. (2018), simple SSL, *i.e* supervised training on a reduced labelled set, has been extensively used to compare different self-supervised methods with latest results getting compelling performances compared to pure SSL algorithms (Chen et al., 2020a,b). While some other works such as Oliver et al. (2018) or Zhou et al. (2018) evaluated SSL methods on transfer learning, they both started with a representation trained on ImageNet. In practice, it can be considered as an upper-bound representation since ImageNet contains all the classes of most of the benchmarks they evaluated on. Therefore, the knowledge of each class was already within the network. Interestingly, both papers showed that transfer to a dataset that shared little to no overlapping classes with the ones the network was pre-trained on lead to poor results because of bias. When self-supervision is not used for pre-training,

it can be directly combined with the SSL algorithm. S4L (Zhai et al., 2019) demonstrated that augmenting semi-supervised models with a loss predicting the rotation of the images, as in RotNet (Gidaris et al., 2018), further improved results over other SSL techniques, even more when the number of labelled images per class is small. This loss was later reused by Berthelot et al. (2019a). Posterior to our work, Chen et al. (2020b) also advocated an automatic use of self-supervised pre-training for SSL. Interestingly, they also report a positive correlation between the size of the network and the final performance on SSL tasks.

Seeking lower dimensional space for clustering. Clustering is known from its early beginning to be sensitive to initialisation and data dimensionality (Beyer et al., 1999). Early works in this domain (Zha et al., 2002; Ding & He, 2004) used Principal Component Analysis (PCA) to find a lower dimensional space where computing distances between samples would be more meaningful. To facilitate learning it is also possible to use hand-crafted features such as Histogram-of-Gradients (HoG) (Chang et al., 2017a), GIST (Hu et al., 2017) or Sobel Filter (Ji et al., 2019). Instead of using hand-crafted kernel functions or features, one could try to use deep networks to learn the image embedding. Hence when using deep networks, self-supervised learning comes in handy as a pre-training step. In particular, auto-encoders (Hinton & Salakhutdinov, 2006) are known to produce low dimensional non-linear mappings. Unsurprisingly, among the first methods that used deep learning to perform clustering, most of them used an auto-encoder during the pre-training step (Tian et al., 2014; Peng et al., 2016). As mentioned in the previous subsection, DEC (Xie et al., 2016) also used self-supervision to learn an image embedding before performing clustering. DCN (Yang et al., 2017) also showed that the space learnt from a stacked auto-encoder was almost linearly separable. This was highlighted by the baseline SAE+KM proposed in their work which performed similar to DEC or DCN. Other than auto-encoders, DEPICT (Ghasedi Dizaji et al., 2017) and ADC (Haeusser et al., 2018) used consistency losses as a way to initialise their network. Recent works such as IMSAT (Hu et al., 2017) or FINCH (Sarfraz et al., 2019) used a representation extracted from ImageNet which, as previously stated, can be seen as an upper bound. However, as the recent results in self-supervised pre-training are reaching supervised training level (Caron et al., 2020; Chen et al., 2020b), it is likely that this gap will be closed soon.

In this thesis we go beyond existing methods in both SSL and clustering. First, we propose self-supervision at the core of our methods when working on dataset with incomplete labelling. This is different from prior works as we explicitly advocate the use of self-supervision in this context. We also consistently demonstrate better results when using self-supervision not only on our methods but on *all* competitors as well. Second, we pushed SSL to the limit of scarce annotations per class in chapter 7. Our method works not only standalone but also as a refinement of existing methods. Different from prior art, our method was the first to consider bigger networks such as ResNet-18 (He et al., 2016) to perform SSL with scarce annotations. Third, in chapter 8 we develop a new way to compare embeddings of a CNN to perform novel category discovery. We show that in this context, it is possible to perform such comparison differently from what was done in past methods (Hsu et al., 2019). Finally chapter 9 improves pseudo-similarity based methods for clustering. We show that instead of comparing outputs at the prediction-level of deep networks, as in DAC (Chang et al., 2017a), we can improve performances by comparing them at the penultimate layer. In addition, we also improve our results thanks to a data augmentation technique that was not considered by past work in clustering.

Chapter 3

Taking visual motion prediction to new heightfields

We start our study of learning visual concepts with less human annotations with intuitive physics.

In this first work, we present a simple yet challenging problem: long term extrapolation and interpolation from visual observations. In particular, this chapter introduces a number of scenarios that are more complex than the flat ones in the literature. We empirically demonstrate that an intermediate tensor representation of internal physics is key when dealing with more complex visual data. We also show that when tasking the model to predict multiple quantities linked by mechanical energy the overall performance of our model as well as others increases. While this work makes use of extensive labelling through simulation or unsupervised tracking (see chapter 4) we will see in subsequent chapters 5 and 6 that learning physics with fewer labels is also possible.

This work was published in the Computer Vision and Image Understanding journal (CVIU) 2019.

Computer Vision and Image Understanding
journal homepage: www.elsevier.com

Taking Visual Motion Prediction To New Heightfields

Sebastien Ehrhardt[a], Aron Monszpart[b], Niloy J. Mitra[b], Andrea Vedaldi[a]

[a]*Department of Engineering Science, Parks road, Oxford, United Kingdom*
[b]*Department of Computer Science, Gower St, London, United Kingdom*

ABSTRACT

While the basic laws of Newtonian mechanics are well understood, explaining a physical scenario still requires manually modeling the problem with suitable equations and estimating the associated parameters. In order to be able to leverage the approximation capabilities of artificial intelligence techniques in such physics related contexts, researchers have handcrafted relevant states, and then used neural networks to learn the state transitions using simulation runs as training data. Unfortunately, such approaches are unsuited for modeling complex real-world scenarios, where manually authoring relevant state spaces tend to be tedious and challenging. In this work, we investigate if *neural networks can implicitly learn physical states of real-world mechanical processes only based on visual data* while internally modeling non-homogeneous environment and in the process enable long-term physical extrapolation. We develop a recurrent neural network architecture for this task and also characterize resultant uncertainties in the form of evolving variance estimates. We evaluate our setup, both on synthetic and real data, to extrapolate motion of rolling ball(s) on bowls of varying shape and orientation, and on arbitrary heightfields using only images as input. We report significant improvements over existing image-based methods both in terms of accuracy of predictions and complexity of scenarios; and report competitive performance with approaches that, unlike us, assume access to internal physical states.

1. Introduction

Animals can make remarkably accurate and fast predictions of physical phenomena in order to perform activities such as navigate, prey, or burrow. However, the nature of the mental models used to perform such predictions remains unclear and is still actively researched Hamrick et al. (2016).

In contrast, science has developed an excellent *formal understanding* of physics; for example, mechanics is nearly perfectly described by Newtonian physics. However, while the constituent laws are simple and accurate, applying them to the description of a physical scenario is anything but trivial. First, the scenario needs to be *abstracted* (*e.g.*, by segmenting the scene into rigid objects, deciding which equations to apply, and estimating physical parameters such as mass, linear and angular velocity, *etc*.). Then, prediction still requires the *numerical integration* of complex systems of equations. It is unlikely that this is the process of mental modeling followed by natural intelligences.

In an effort to develop models of physics that are more suitable for artificial intelligence, in this work, we ask whether a representation of the physical state of a mechanical system can be learned *implicitly* by a neural network, and whether this can be used to perform accurate predictions efficiently (i.e., extrapolating to predict future events). To this end, we propose a new learnable representation with several important properties. First, the representation is not handcrafted, but rather *automatically induced from visual observations* using supervision from known physical quantities such as positions and angular velocities. Second, the representation is *distributed* and can model physical interactions of objects with *complex surrounding*, such as irregularly-shaped ground. Third, despite its distributed nature, the representation can model a number of *interacting discrete objects* such as colliding balls, without the need of ad-hoc components such as collision detection subnetworks. Fourth, since physical predictions integrate errors over time and are thus inherently ambiguous, the

e-mail: hyenal@robots.ox.ac.uk (Sebastien Ehrhardt), aron.monszpart.12@cs.ucl.uk (Aron Monszpart), n.mitra@cs.ucl.ac.uk (Niloy J. Mitra), vedaldi@robots.ox.ac.uk (Andrea Vedaldi)

representation produces *robust probabilistic predictions* which model such ambiguity explicitly. Finally, through extensive evaluation, we show that the representation performs well for both *extrapolation* and *interpolation* of mechanical phenomena.

Our paper is not the first that looks at learning to predict mechanical phenomena using deep networks. In particular, inducing a physical representation automatically from visual data and handling object interactions were explored in (Fragkiadaki et al., 2016; Watters et al., 2017). However, in this paper we propose a method that combines these benefits, with the main technical novelty of a distributed tensor representation for the physical state. The latter enables us to consider more complex environments.

Earlier, the recent Neural Physics Engine (NPE) of Chang et al. (2017) uses a neural network to learn the state transition function of mechanical systems. Differently from ours, their state is handcrafted and includes physical parameters such as positions, velocities, and masses of rigid bodies. While NPE works well, it still requires to abstract the physical system manually, by identifying the objects and their physical parameters, and by explicitly integrating such parameters. In practice, this requires an extensive knowledge of the environment that, in turn, can bring in more estimation errors (Yu et al., 2016). In contrast, our abstractions are entirely induced from external observations of object motions. Hence, our system implicitly discovers any hidden variable or state required to perform tasks such as *long-term physical extrapolation* in an optimal manner. Furthermore, the integration of physical parameters over time is also implicit and performed by a recurrent neural network architecture. This is needed since the nature of the internal state is undetermined; it also has a major practical benefit as, as we show empirically, the system can be trained to not only extrapolate physical trajectories, but also to interpolate them. Remarkably, interpolation is still obtained by computing the trajectory in a feed-forward manner, from the first to the last time step, using the recurrent model.

Another significant difference with NPE is in the fact that our system uses *visual observations* to perform its predictions. In this sense, the work closest to ours is the *Visual Interaction Networks* (VIN) of Watters et al. (2017), which also use visual input for prediction. However, our system is significantly more advanced as it can model the interaction of objects with complex and irregular terrain. We show empirically that VIN is not very competitive in our more complex experimental setting.

There are also several aspects that we address for the first time in this paper. Empirically, we push our model by considering scenarios beyond the 'flat' ones tackled by most recent papers, such as objects sliding and colliding on planes, and look for the first time at the case of ball(s) rolling on non-trivial 3D shapes (e.g., bowls of varying shape and orientation, or terrains modeled as arbitrary heightfields), where both linear and angular momenta are tightly coupled. Our method is evaluated on both *synthetic and real* data using the Roll4Real dataset of Ehrhardt et al. (2018). We also increased the complexity of the task by training models that simultaneously estimate *positions and angular velocity*. Furthermore, since physical extrapolation is inherently ambiguous, we allow the model to explicitly estimate its prediction uncertainty by estimating the variance of a Gaussian observation model. We show that this modification further improves the quality of long-term predictions. While our work builds on previous research of Ehrhardt et al. (2017a) and Ehrhardt et al. (2017b), in this paper we propose a much more complex set of experiments including the irregularly shaped heightfield and multiple balls experiment as well as stronger baselines that highlight the performances of our models. Furthermore, a more careful study of the various results presented has been conducted and exhaustively discussed.

The rest of the paper is organized as follows. The relation of our work to the literature is discussed in section 2. The detailed structure of the proposed neural networks is given and motivated in section 3. These networks are tested on a large dataset of simulated physical experiments described in section 4 and extensively evaluated and contrasted against related works in section 5. We conclude by discussing current limitations and directions for future investigation in section 6.

2. Related Work

We address the problem of training deep neural networks that can perform long-term predictions of mechanical phenomena while learning the required physical laws implicitly, via empirical and visual observation of the motion of objects. This research is thus related to a number of recent works in various machine learning sub-areas, discussed next.

Learning intuitive physics. Battaglia et al. (2013) are one of the first to consider 'intuitive' physical reasoning; their aim is to answer simple qualitative questions related to rigid body processes, such as determining whether a certain tower of blocks is likely to fall or not. They approach the problem by using a sophisticated physics engine that incorporates all required knowledge about Newtonian physics *a-priori*. More recently, Mottaghi et al. (2016) used static images and a graphics rendering engine (Blender) to predict motion and forces from a single RGB image. Motivated by the recent success of deep learning for image analysis (*e.g.*, Krizhevsky et al. (2012)), they trained a convolutional neural network to predict such quantities and used it to produce a "most likely motion," rendering it using a traditional computer graphics pipeline. With a similar motivation, Lerer et al. (2016) and Li et al. (2017) also applied deep networks to predict the stability of towers of blocks purely from images. These approaches demonstrated that such networks can not only predict instability, but also pinpoint the source of such instability, if any. Other approaches such as Agrawal et al. (2016) or Denil et al. (2016) have attempted to learn intuitive physics of objects through manipulation; however, their models did not aim to capture the underlying dynamics of the systems.

Learning physics. The work by Wu et al. (2015) and its extension Wu et al. (2016) propose methods to learn physical properties of scenes and objects. Wu et al. (2015) use an MCMC-sampling based approach that assumes complete knowledge of the physical equations necessary to estimate physical parameters. In Wu et al. (2016), a deep learning based approach was used instead of MCMC, albeit still explicitly encoding physics in a

simulator. Physical laws were also explicitly incorporated in the model by Stewart and Ermon (2017) to predict the movement of a pillow from unlabelled data. Their method was, however, only applied to a fixed small number of future frames. Work of Yu et al. (2016) proposing a high fidelity dataset for planar pushing reveals that what might be thought of as a simple task, to push an unknown object to a desired position, remains a challenging task in robotics, and is generally better explained by stochastic models than by estimating and modelling the physical world.

The research performed by Battaglia et al. (2016) and Chang et al. (2017) focused on dynamics and attempted to partially substitute the physics engine with a neural network that captures a selection of relevant physical laws. Both approaches were able to use such networks to accurately predict updates for the physical state of the world. Although results are plausible and promising, Chang et al. (2017) suggest that long-term predictions remain difficult. Furthermore, in both approaches, their neural networks only predict instantaneous updates of physical parameters that are then explicitly integrated. In contrast, in this work propagation is implicit and applies a recurrent neural network architecture to an implicit representation of the world.

Closer to our approach, Fragkiadaki et al. (2016) and Watters et al. (2017) attempted to learn an internal representation of the physical world from images. In addition to observing images, it is also possible to *generate* them as Fragkiadaki et al. (2016) learn to perform long-term extrapolation more successfully. The work of Fragkiadaki et al. (2016) particularly differs from ours in this last point and the nature of its internal representation (vector representation in their work *vs* tensor in ours) which, as demonstrated in 5.2, is essential for our method. Similarly Wu et al. (2017) also used a physics engine and a renderer to make future predictions. In both cases, image generation can be seen as a constraint that avoids the over time degeneration of the internal representation of dynamics. However, these approaches need exhaustive and exact knowledge of every object and the environment, information generally not accessible in real life scenarios. The work of Watters et al. (2017) extends the Interaction Network by Battaglia et al. (2016) to propagate an implicit representation of the dynamics of objects, obtaining a Visual Interaction Network (VIN). While their approach is the closest to ours, it has various limitations including not modeling the interaction with complex environments and the relatively small size of the input images. The Predictron by David et al. (2016) also propagates a tensor state, but suffers from the same drawbacks. Ehrhardt et al. (2017b) showed, how long-term extrapolation models can be trained for one object moving on smooth analytic surfaces, such as ellipsoids.

Approximating physics for plausible simulation. Several authors focused on learning to perform plausible physical predictions, for example to generate realistic future frames in a video Tompson et al. (2016); Ladický et al. (2015), or to infer rigid body collision parameters from monocular videos Monszpart et al. (2016). In these approaches, physics-based losses are used to learn plausible yet not necessarily accurate results, which may be appropriate for tasks such as rendering and animation. Battaglia et al. (2016) also use a loss that captures the concept of energy conservation. The latter can be seen as a way to incorporate knowledge about physics *a-priori* into the network, which differs from our goal of learning any required physical knowledge from empirical observations.

Learning dynamics. Physical extrapolation can be performed without integrating physical equations explicitly. For example, LSTMs Hochreiter and Schmidhuber (1997) were used to make accurate long-term predictions in human pose estimation Villegas et al. (2017) and in simulated environments Oh et al. (2015); Chiappa et al. (2017). Propagation can also be done using simpler convolutional operators; Xue et al. (2016), in particular, used these to generate possible future frames given a single static image and De Brabandere et al. (2016) applied it to the moving MNIST dataset for long-term prediction. The work by Ondruska and Posner (2016) and Ehrhardt et al. (2017a) also showed that an internal representation of dynamics can be propagated through time using a simple deep recurrent architecture. Greff et al. (2017) demonstrated that information about dynamics can be used to efficiently cluster different observed shapes. Our work builds on their success, and propagates a tensor-based state representation instead of a vector-based one. Using spatial convolutional operators allows for knowledge to be stored and propagated locally *w. r. t.* the object locations in the images.

3. Method

In this section, we propose a novel neural network model to make predictions about the evolution of a mechanical system from visual observations of its initial state. In particular, this network, summarized in Fig. 1, can predict the motion of one or more rolling objects accounting for variations in the 3D geometry of the environment.

Formally, let y_t be a vector of physical quantities that we would like to predict at time t, such as the position of one or more objects. Physical systems satisfy a Markov condition, in the sense that there exists a state vector h_t such that (i) measurements $y_t = g(h_t)$ are a function of the state and (ii) the state at the next time step $h_{t+1} = f(h_t)$ depends only on the current value of the state h_t. Uncertainty in the model can be encoded by means of observation $p(y_t|h_t)$ and transition $p(h_{t+1}|h_t)$ probabilities, resulting in a hidden Markov model.

State-only methods, such as the Neural Physics Engine (NPE) by Chang et al. (2017) start from an handcrafted definition of the state h_t. For instance, in order to model a scenario with two balls colliding, one may choose h_t to contain the position and velocity of each ball. In this case, the observation function g may be as simple as extracting the position components from the state vector. It is then possible to use a neural network ϕ to approximate the transition function f. In particular, Chang et al. (2017) suggest that it is often easier for a network to predict a rate of change $\Delta_t = \phi(h_t)$ for some of the physical parameters (*e.g.*, the balls' velocities), which can be used to update the state using a hand-crafted integrator $h_{t+1} = \tilde{f}(h_t, \Delta_t)$.

While this approach works well, there are several limitations. First, even if the transition function is learned, the state h_t is defined by hand. Even in the simple case of the colliding balls, the choice of state is ambiguous; for example, one could include in the state not only the position and velocity of the balls,

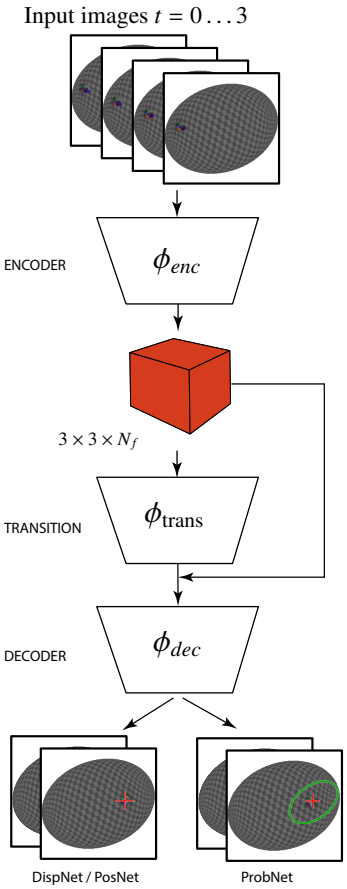


Fig. 1: **Overview of our proposed pipeline.** The first four images of a sequence first pass through a partially pre-trained feature encoder network to build the concept of physical state. It then recursively passes through a transition layer to produce long-term predictions about the future states of the objects. It is then decoded to produce state estimates. While our *DispNet* and *PosNet* models are trained to regress the next states, the *ProbNet* model trained with the log-likelihood loss is also able to handle the notion of uncertainty thanks to its extended state space. Note here that only one object is considered, extension for multiple objects is discussed in section 3.4.

but also their radius, mass, elasticity, friction coefficients, *etc*. Learning the state as well has the significant benefit of making such choices automatic. Second, training a transition function requires knowledge of the state values, which may be difficult to obtain except in the case of simulated data. Third, in order to use such a system to perform predictions, one must know the initial value of the state h_0 of the system, whereas in many applications one would like to start instead from *sensory inputs* x_t such as images Fragkiadaki et al. (2016).

We propose here an approach to address these difficulties. We assume that the state h_t is a *hidden variable*, to be determined as part of the learning process. Since the h_t cannot be observed, the transition function $h_{t+1} = f(h_t)$ cannot be learned directly as in the NPE. Instead, state and transitions must be inferred jointly as a *good explanation* of the observed physical measurements y_t. Any integrator involved in the computation of the transition function is implicitly moved *inside the network*, which is a recurrent neural network architecture. In our experiments (section 5), we show that the added flexibility of learning an internal state representation and its evolution automatically allows the system to scale well to the complexity of the physical scenario.

Since the evolution of the state h_t cannot be learned by observing measurements y_t in isolation, the system is supervised using sequences $\mathbf{y}_{[0,T)} = (y_0, \ldots, y_{T-1})$ of observations. This is analogous to a Hidden Markov Model (HMM), which is often learned by maximizing the likelihood of the observation sequences after marginalizing the hidden state.[1] As an alternative learning formulation, we propose instead to consider the problem of long-term predictions starting from an initial set of observations. Not only this is more directly related to applications, but it has the important benefit that predictions can be performed equally well from initial observations of the physical quantities y_t *or* of some other sensor reading x_t, such as images.

Our system is thus based on learning three modules: (i) an *encoder* function that estimates the state $h_t = \phi_{\text{enc}}(\mathbf{x}_{(t-T_0,t]})$ from the T_0 most recent sensor readings (alternatively $h_t = \phi_{\text{enc}}(\mathbf{y}_{(t-T_0,t]})$ can use the T_0 most recent physical observations); (ii) a *transition* function $h_{t+1} = \phi_{\text{trans}}(h_t)$ that evolves the state through time; and (iii) a *decoder* function that maps the state h_t to a physical observation $y_t = \phi_{\text{dec}}(h_t)$, and in some case an uncertainty associated. The rest of the section discusses the three modules, encoder, transition, and decoder maps, as well as the loss function used for training. Further technical details can be found in section 5.

3.1. Encoder Map: from images to state

The goal of the encoder map is to take T_0 consecutive video frames observing the initial part of the object motion and to produce an estimate $h_0 = \phi_{\text{enc}}(\mathbf{x}_{(-T_0,0]})$ of the initial state of the physical system. In order to build this encoder, we follow Fragkiadaki et al. (2016) and concatenate the RGB channels of the T_0 images in a single $H_i \times W_i \times 3T_0$ tensor. The latter is passed to a convolutional neural network ϕ_{enc} outputting a feature tensor $s_0 \in \mathbb{R}^{H \times W \times C}$, used as internal representation of the system's state. Note that this representation is spatially distributed and differs from the concentrated vector representation of the VIN of Watters et al. (2017). In the experiments, we will show the advantage of using a tensorial representation in modeling complex environments. We also augment our tensor representation with a state vector $p_t \in \mathbb{R}^n$, so that the state is the pair $h_t = (s_t, p_t)$. In deterministic cases, $n = 2$ and p_t is the 2D projection of the object's location on the image plane. For multiple objects (see section 3.4) this state is computed for each object independently.

3.2. Transition Map: evolving the state

The state h_t is evolved through time by learning the transition function $\phi_{\text{trans}} : h_t \mapsto h_{t+1}$. Since the initial state h_0 is obtained from the encoder map, the state at time t can be written as, $h_t = \phi^t_{\text{trans}}(\phi_{\text{enc}}(\mathbf{x}_{(-T_0,0]}))$.

More in detail, the distributed state component s_t is updated by using a convolutional network $s_{t+1} = \phi_s(s_t)$. The concentrated

[1] Formally, a Markov model is given by $p(\mathbf{y}_{[0,T)}, \mathbf{h}_{[0,T)}) = p(h_0)p(y_0|h_0)\prod_{t=0}^{T-2} p(h_{t+1}|h_t)p(y_{t+1}|h_{t+1})$; traditionally, p can be learned as the maximizer of the log-likelihood $\max_p E_{\mathbf{y}}[\log E_{\mathbf{h}}[p(\mathbf{y}, \mathbf{h})]]$, where we dropped the subscripts for compactness. Learning to interpolate/extrapolate can be done by considering subsets $\bar{\mathbf{y}} \subset \mathbf{y}$ of the measurements as given and optimizing the likelihood of the conditional probability $\max_p E_{\mathbf{y}}[\log E_{\mathbf{h}}[p(\mathbf{y}, \mathbf{h}|\bar{\mathbf{y}})]]$.

component p_t is updated incrementally as $p_{t+1} = p_t + \phi_p(s_t)$, where $\phi_p(s_t)$ is estimated using a single layer perceptron regressor from the distributed representation. Combined, the state update can be written as,

$$(s_{t+1}, p_{t+1}) = \phi_{\text{trans}}(s_t, p_t) = (\phi_s(s_t), p_t + \phi_p(s_t)).$$

Inspired by the work of Watters et al. (2017), we also consider an alternative architecture where p_t is estimated directly from s_t rather than incrementally. In order to do so, the location x and y of each pixel is appended as feature channels $C + 1$ and $C + 2$ of the distributed state tensor s_t, obtaining an augmented tensor $\text{aug}_{xy}(s_t)$. Then the object's position p_t is estimated by a two-layer perceptron $p_t = \phi_p(\text{aug}_{xy}(s_t))$.

3.3. Decoder Map: from state to probabilistic predictions

For deterministic models, the projected object position p_t is part of the neural network state, the decoder map $\hat{y}_t = \phi_{\text{dec}}(s_t, p_t) = p_t$ simply extracts and returns that part of the state. Training optimizes the average L^2 distance between ground truth y_t and predicted $\hat{y}_t$ positions $\frac{1}{T}\sum_{t=0}^{T-1} \|\hat{y}_t - y_t\|^2$.

In addition to this simple scheme, we also consider a more robust variant based on probabilistic predictions. In fact, the extrapolation error accumulates and increases over time, and the L^2-based loss may be dominated by outliers, unbalancing learning. Hence, we modify the model to explicitly and dynamically express its own prediction uncertainty by outputting the mean and variance (μ_t, Σ_t) of a bivariate Gaussian observation model. The L^2 loss is thus replaced with the negative log likelihood $-\frac{1}{T}\sum_{t=0}^{T-1} \log \mathcal{N}(y_t; \mu_t, \Sigma_t)$ under this model.

In order to estimate the Gaussian parameters μ_t and Σ_t, we extend the state component $p_t = (\mu_t, \lambda_{1,t}, \lambda_{2,t}, \theta_t)$ to include both the mean as well as the eigenvalues and rotation of the covariance matrix $\Sigma_t = R(\theta_t)^\intercal \,\text{diag}(\lambda_{1,t}, \lambda_{2,t}) R(\theta_t)$. In order to ensure numerical stability, eigenvalues are constrained to be in the range $[0.01, 100]$ by setting them as the output of a scaled and translated sigmoid $\lambda_{i,t} = \sigma_{\lambda,\alpha}(\beta_{i,t})$, where $\sigma_{\lambda,\alpha}(z) = \lambda/(1 + \exp(-z)) + \alpha$. In the following, we will refer to this method as *ProbNet*, whereas the other method estimated displacement without uncertainty will be referred to as *DispNet*. Table 1 summarize the different methods and their specificity.

3.4. Extension to multiple objects

We now consider how the model described above can be extended to handle multiple interacting objects. This is more challenging as it requires to handle complex object interactions such as collisions.

In order to do so, for each object o_i, $i = 1, \ldots, N_{\text{objects}}$ we consider a separate copy of the distributed state tensor $s_t^{o_i}$ (hence the overall state is $s_t = (s_t^{o_1}, \ldots, s_t^{o_{N_{\text{objects}}}})$). The encoder network ϕ_{enc} is thus modified to output a $H \times W \times N_{\text{objects}}C$ tensor. It is then split along the third dimension to produce $H \times W \times C$ tensor for each of the $N_{objects}$. We order objects *w. r. t.* their color so that each feature is always responsible for the same object identified by its color. We recall here that this extension studies the ability of handling collisions of our model without any explicit module. We aim in the future to build more object agnostic representation.

The input of the transition module is also modified to take into account the interaction between objects. Focusing on an object o_f with state $s_t^{o_f}$, the update is written as

$$s_{t+1}^{o_f} = \phi_s\left(s_t^{o_f}, \sum_{i \neq f} s_t^{o_i}\right)$$

where the second argument is the sum of the state subtensors for all *other* objects. Since the function ϕ_s is the same for all objects o_f, this ensures that object interactions are symmetric and commutative. Note that, as opposed to methods such as Chang et al. (2017), no explicit collision detection module is implemented here. Instead, handling collisions is left to the discretion of the network.

With this modification, the transition subnetwork is illustrated in Fig. 2. The rest of the pipeline is essentially the same as before and is applied independently to each object. The same network parameters are used for each application of a module regardless of the specific object.

Table 1: **Neural network variants.**

Name	p_t regression	p_{t+1}	output and loss
DispNet	incremental	$p_t + \phi_p(s_t)$	deterministic
ProbNet	incremental	$p_t + \phi_p(s_t)$	probabilistic
PosNet	direct	$\phi_p(s_t)$	deterministic

4. Experimental Setup

Experiments were conducted on both real and synthetic datasets. In the synthetic experiments (Fig. 3), we consider two physical scenarios: spheres rolling on a 3D surface, which can be either a semi-ellipsoid with random parameters or a continuous randomized heightfield. When the semi-ellipsoid is isotropic (*i.e.* a hemisphere) we refer to it as 'Hemispherical bowl', and in the more general case as 'Ellipsoidal bowl' (see Table 2), whereas the heightfield scenario is referred to as 'Heightfield.'

4.1. Hemispherical bowl and Ellipsoidal bowl scenarios

The symbol $\mathbf{p} = (p_x, p_y, p_z) \in \mathbb{R}^3$ denotes a point in 3D space or a vector (direction). The camera center is placed at location $(0, 0, c_z)$, $c_z > 0$ and looks downward along vector $(0, 0, -1)$

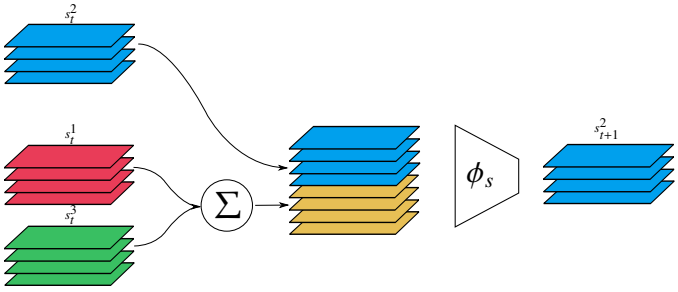


Fig. 2: **Multiple object extension.** For each object (here object 2) we concatenate the state of this object with the addition of the other objects features. We then give this tensor to the module ϕ_s to obtain our new state s_{t+1}^2

using orthographic projection, such that the point (p_x, p_y, p_z) projects to pixel (p_x, p_y) in the image.

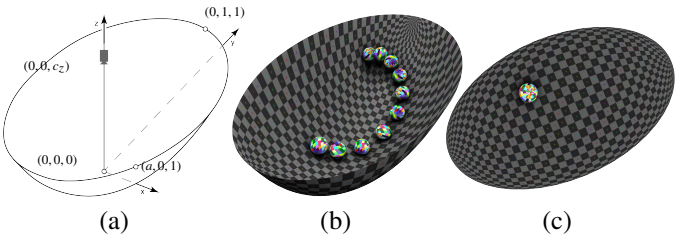


Fig. 3: **Problem setup.** We consider the problem of understanding and extrapolating mechanical phenomena with recurrent deep networks. (a) Experimental setup: an orthographic camera looks at a ball rolling in a 3D bowl. (b) Example of a 3D trajectory in the 3D bowl simulated using Blender 2.78's OpenGL renderer. (c) An example of a rendered frame in the 'Ellipsoidal bowl' experiment that is fed to our model as input.

Thus, the 'Ellipsoidal bowl' is the bottom half of an ellipsoid of equation $x^2/a^2 + y^2 + (z-1)^2 = 1$ with its axes aligned to the xyz axes and its lowest point corresponding to the origin. For the 'Ellipsoidal bowl' scenario, the ellipsoid shape is further varied by sampling $a \in U[0.5, 1]$ for the ($a = 1$ for the 'Hemispherical bowl' scenario) and by rotating the resulting shape randomly around the z-axis. Both 'Hemispherical bowl' and 'Ellipsoidal bowl' are rendered by mapping a checker board pattern to their 3D surface (to make it visible to the network).

The rolling object is a ball of radius $\rho \in \{0.04, 0.225\}$. The ball's center of mass at time t is denoted as $\mathbf{q}^t = (q_x^t, q_y^t, q_z^t)$, which, due to the orthographic projection, is imaged at pixel (q_x^t, q_y^t). The ball has a fixed multi-color texture attached to its surface, so it appears as a painted object. The texture is used to observe the object rotation. We study the impact of being able to visually observe rotation by re-rendering the single ball experiments with a uniform white color (see Table 3). In the multi-object experiments, instead, each ball has a constant, distinctive diffuse color (intensity 0.8) with Phong specular component (intensity 0.5). We initially position the ball at angles (θ, ϕ) with respect to the the bowl center, where the elevation θ is uniformly sampled in the range $\theta \in U[-9\pi/10, -\pi/2]$ and the azimuth $\phi \in U[-\pi, \pi]$. The minimum elevation is set to $-9\pi/10$ to avoid starting the ball at the bottom of the bowl. Due to friction, at the end of each experiment the ball rests at the bottom of the bowl.

The initial orientation of the ball (relevant for the multi-colored texture) is obtained by uniformly sampling its xyz Euler angles in $[-\pi, \pi]$. The ball's initial velocity $\mathbf{v}$ is obtained by first sampling v_x, v_y uniformly in the range $U[5, 10]$, assigning each of v_x, v_y a random sign ($\sim 2\mathcal{B}(0.5) - 1$), and then by projecting the vector $(v_x, v_y, 0)$ to be *tangential* to the bowl's surface. In the multi-object 'Ellipsoidal bowl' scenario, in order to achieve more interesting motion patterns, the magnitude of the initial velocities is set uniformly in the range $U[10, 15]$; if, after simulation, a ball leaves the bowl due to a collision or excessive initial velocity, the scene is discarded. Sequences are recorded until all objects stop moving. Short sequences (less than 250 frames) are discarded as well. The average angular velocity computed over all 'Bowl' scenes was 5.94 radian/s.

Note that, while some physical parameters of the ball's state

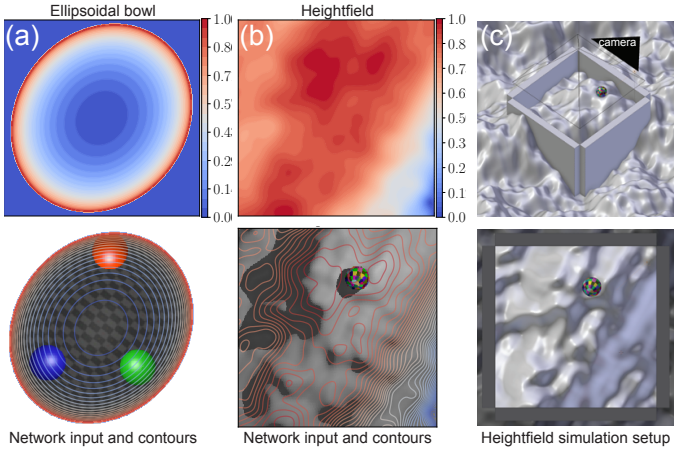


Fig. 4: **Experimental setups.** (a) 'Ellipsoidal bowl' experiment setup, depth map on the top, network input with isocontours at the bottom. We create the dataset by varying the ellipsoid's main axis ratio and orientation, and the starting position and velocity of the balls. (b-c) 'Heightfield' rendering setup. Each sequence is generated using a random translation and rotation of the fixed heightfield geometry. Walls ensure the automatically generated sequences are long enough. A randomly positioned area light presents additional generalization challenges to the network.

are included in the *observation* vector $\mathbf{y}^\alpha_{[-T_0,T)}$, these are *not* part of the state h of the neural network, which is inferred automatically. The network itself is tasked with *predicting* part of these measurements, but their meaning is not hardcoded.

Simulation details

For efficiency, we extract multiple sub-sequences $\mathbf{x}^\alpha_{[-T_0,T)}$ form a single longer simulation (training, test, and validation sets are however completely independent). The simulator runs at 120fps for accuracy, but the data is subsampled to 40fps. We use Blender 2.78's OpenGL renderer and the Blender Game Engine (relying on Bullet 2 as physics engine). The ball is a textured sphere with unit mass. The simulation parameters were set as: max physics steps = 5, physics substeps = 3, max logic steps = 5, fps = 120. Rendering used white environment lighting (energy = 0.7) and no other light source in the 'Hemispherical bowl' case, environment energy = 0.2, and a spotlight at the location of the camera in the 'Ellipsoidal bowl' case. We used 70% the data for training, 15% for validation, and 15% for test, 12500 sequences in the 'Hemispherical bowl'/'Ellipsoidal bowl' experiments and 6400 in the 'Heightfield' case. During training, we start observation at a random time while it is fixed for test. The output images were stored as 256×256 color JPEG files. For multiple objects in the ellipsoid experiment, we set the elasticity parameter of the balls to 0.7 in order to get a couple of collisions before they settle in the middle of the scene.

4.2. *Heightfield scenario*

An important part of our experiments involve randomly generated continuous heightfields. Long-term motion prediction on random heightfields represent a tougher challenge, since solely observing the motion of the object at the beginning of

the sequence does not contain enough information for successful mechanical predictions. In contrast to the 'Ellipsoidal bowl' cases, where the 2D shape that the container occupies in the image is theoretically enough to infer the analytical shape of the local surface at any future 3D point of interest, in the 'Heightfield' case the illumination conditions of the surface have to be parsed. Furthermore, a more elaborate understanding about the interaction between surface and 3D rolling motion has to be developed.

Similar to the ellipsoid cases, we generate randomized sequences of a ball rolling on a random (heightfield) surface. We approximate random heightfields by generating a large (8×8) Improved Perlin noise texture and applying it as a displacement map to a highly tessellated plane. For each scene, we uniform randomly rotate and translate the plane so that a different part (2.5×2.5) of the heightfield is visible under the static camera. In order to generate motion sequences of enough length for long-term extrapolation, we also surround the camera frustum with perfectly elastic walls (see Fig. 4c). The noise texture has a scale parameter, which we vary between 0.7 (fairly planar) and 0.2 resulting in high curvature surfaces that have holes comparable with the ball diameter. We set the surface elasticity to 0 in order to encourage the balls to roll and not bounce. The initial placement of the ball, similarly to the bowl case, is drawn from a 2D uniform distribution. Then, we use sphere tracing to push the ball onto the surface from the camera plane. We add a small random initial velocity ($U[2, 4]$), and similarly to the 'Hemispherical bowl' case, we project the initial velocity onto the local surface normal. The average angular velocity computed over all 'Heightfield' scenes was 2.8 radian/s. The surface is lit with a small (0.1×0.1) area light from a random location. We draw the 2D position of the light as $x, y \sim (2\mathcal{B}(0.5) - 1)(U[1, 1.5] \times U[1, 1.5])$, with a fixed camera height $z = 2$.

4.3. Real data

Additonally, we experimented on real data. We evaluated our methods on the Roll4real dataset by Ehrhardt et al. (2018). The dataset consists of 1118 short 256×256 videos containing one or two balls rolling on three types of terrains: a flat pool table PoolR, a large ellipsoidal 'bowl' BowlR, and an irregular height-field HeightR. More specifically, there are 151 videos (avg. 99 frames/video) for the PoolR dataset with one ball; 216 videos (522 frames/video). For the BowlR dataset with one ball; 543 videos (avg. 356 frames/video) for the HeightR dataset with one ball; and 208 videos (avg. 206 frames/video) for the HeightR dataset with two balls. More details about the dataset and the way to obtain ground-truth annotations can be found in Ehrhardt et al. (2018).

5. Results and Discussions

5.1. Baselines

(i) Least squares fit. We compare the performance of our methods to two simple least squares baselines: Linear and Quadratic. In both cases, we fit least squares polynomials to the screen-space coordinates of the first $T = 10$ frames, which are not computed but given as inputs. The polynomials are of first and second degree(s), respectively. Note, that being able to observe the first 10 frames is a large advantage compared to the networks, which only see the first $T_0 = 4$ frames.

(ii) NPE. The NPE method and its variants were trained using available online code. We used the same training procedure as reported in Chang et al. (2017). Additionally, we added angular velocities as input and regressed type of parameter. In the case of the Ellipsoidal bowl, both scaling and bowl rotation angle are also given as input to the networks. In this case NPE's method carries forward the estimated states via the network.

(iii) V-LSTM model. Inspired by the models of Fragkiadaki et al. (2016), we developed an LSTM architecture as a baseline. The architecture is similar to the one in Fragkiadaki et al. (2016), as it reuses the exact same truncated pre-trained AlexNet encoder, and the same LSTM and decoder architecture with the following **two differences**: First, we do not regenerate images to produce a new input for the LSTMs, we rather used the last output of the LSTM. Second, the decoder only produces the next state estimate and not the 20 next ones to make it a fair comparison to our models. As the multiple ball version in the original paper required centering the frame of each independent object and regenerating images at every time step, we considered a simple reimplementation with one V-LSTM network for each of the individual objects. Thus we can take into account the entire frame without any need to regenerate images.

(iv) VIN and IN From State (IFS). Finally, we used VIN network and its state variant IN From State from Watters et al. (2017). IFS is essentially a version of VIN where the propagation mechanism is the same but the first state vector is not deduced from visual observation but given as ground truth position and velocity as in the NPE. The VIN network uses downscaled 32×32 images. Both networks use training procedures as reported in Watters et al. (2017) with the exceptions that for IFS the learning rate was updated using our method (see section 5.2) and we rely on the first 4 states and 16 rolled out steps. As with NPE, angular velocity was also added to IFS input and regressed parameters. Scaling and rotation angle of the bowl were also given as input to the network in Ellipsoidal bowl experiment. Note that VIN and our models work with images as direct observation of the world rather than perfect states, which represents a much more difficult problem whilst yielding a more general applicability. Physical properties are then deduced from the observations and integrated through our Markov model. Thus, these methods do not need a simulator to estimate parameters of the physical worlds (such as scaling and rotation angle) and can be trained on changing environments without requiring additional external measurements of the underlying 3D spaces.

5.2. Results

Implementation details. The encoder network ϕ_{enc} is obtained by taking the ImageNet-pretrained VGG16 network of Simonyan and Zisserman (2015) and retaining the layers up to `conv5` (for an input image of size $(H_i, W_i) = (128, 128, 3)$ this results in a $(8, 8, N_f = 512)$ state tensor s_t). In the 3 balls experiments, we

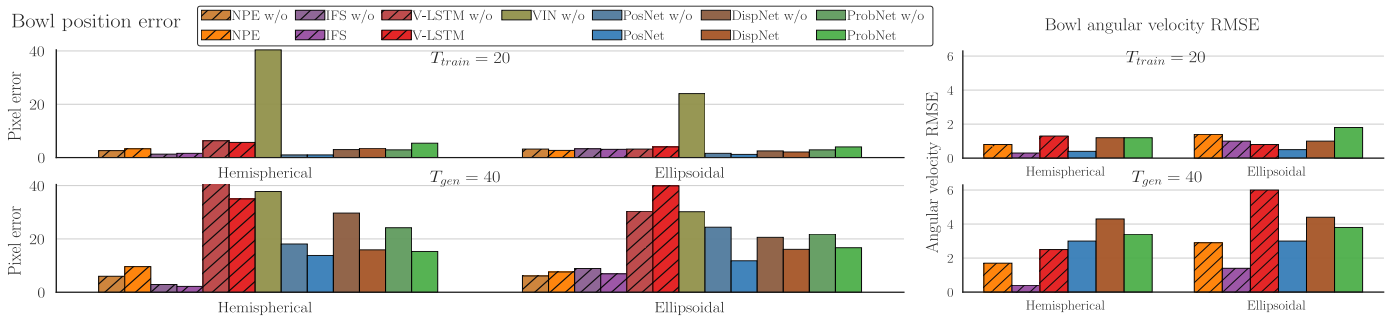


Fig. 5: **Errors in bowls.** Pixel errors and angular velocity RMSE in radian/s (first two columns of Table 2). Our method performs comparably to state based methods, which use ground truth state information for initialization compared to ours, which operates with *visual* input. Hatched denotes non-visual input (*i.e.* direct access to physical states).

replaced the last `conv5` layer with a convolutional layer of output 256×3 channels. Object features are thus obtained by splitting this last tensor along the channel dimension into $(8, 8, N_f = 256)$ state tensor per object. The filter weights of all layers except `conv1` are retained for fine-tuning on our problem. The `conv1` is reinitialized as filters must operate on images with $3T_0$ channels. The transition network $\phi_s(s_t)$ uses a simple chain of two convolution layers[2]with 256 and N_f filters respectively, of size 3×3, stride 1, and padding 1 interleaved by a ReLU layer.Network weights are initialized by sampling from a Gaussian distribution. Additionally, angular velocity is always regressed from the state s_t using a single layer perceptron.

Training uses a batch size of 50 using the first T_{train} positions and angular velocity (or only position when explicitly mentioned) of each video sequence using RMSProp by Tieleman and Hinton (2012). We start with a learning rate of 10^{-4} and decrease it by a factor of 10 when no improvements of the loss have been found after 100 consecutive epochs. Training is halted when the loss has not decreased after 200 successive epochs; 2,000 epochs were found to be usually sufficient for convergence. In every case the loss is the sum of the L^2 angular velocity loss and either L^2 position errors (*PosNet*, *DispNet*) or likelihood loss (*ProbNet*) (see section 3.3). We omit the angular loss, when angular velocity is not regressed (labelled as "* *w/o ang. vel.*" in the tables).

Since during the initial phases of training the network is very uncertain, the model using the Gaussian log-likelihood loss was found to get stuck on solutions with very high variance $\Sigma(t)$. To address this, we added a regularizer $\lambda \sum_t \det \Sigma(t)$ to the loss, with $\lambda = 0.01$.

In all our experiments we used Tensorflow (Abadi et al. (2015)) r1.3 on a single NVIDIA Titan X GPU.

5.2.1. Extrapolation

(i) Experiments using a single ball. Table 2 compares the baseline predictors and the eight networks on the task of long term prediction of the object trajectory. All methods observed only the first $T_0 = 4$ inputs (either object states or simply image frames) except for the linear and quadratic baselines, and aimed to extrapolate the trajectory to $T_{\text{gen}} = 40$ time steps. Predictions are "long term" relative to the number of inputs $T_0 \ll T_{\text{gen}}$. Note also that during training networks only observe sequences of up to $T_{\text{train}} \leq T_{\text{gen}}$ frames; hence, the challenge is not only to extrapolate physics, but to generalize *beyond* extrapolations observed during training.

Quantitative evaluation. Table 2 reports the average errors at time $T_{\text{train}} = 20$ and $T_{\text{gen}} = 40$ for the different estimated parameters. Our methods outperform state-only approaches for predictions of up-to T_{train} steps. For example, PosNet has a pixel error of 1.0/1.2/6.8 in the Hemispherical/Ellipsoidal/Heightfield scenarios vs 3.3/2.7/10.9 of NPE, 1.6/3.1/8.7 of IFS. This is non-trivial as our networks know nothing about physical laws a-priori, and observe the world through images rather than being given the initial ground-truth state values. On the other hand, our methods can, through images, better observe and hence model the underlying environments. The gap in the heightfield results, in particular, shows the value in observing the environments in this manner as we constantly out-perform state-only methods. Our methods also shown to make significantly better predictions compared to the other visual competitors. For instance, V-LSTM was unable to match the strong performance of our networks (pixel errors are 5.7/4.0/8.8 in the Hemispherical/Ellipsoidal/Heightfield scenarios respectively) highlighting the advantage of a spatially distributed tensor state representation as opposed to a vector one. As for the VIN network, it failed to be able to model interactions between the object and its environment and performed poorly even on training regimes (40.4/24.0/42.6 respectively).

All methods can perform arbitrary long predictions. Our networks, which are only trained to predict the first T_{train} positions, are still competitive with state-only methods (which only predict a transition function and hence implicitly generalize to arbitrarily lengths) even when predictions are generalized to T_{gen} steps. In particular, while performances around T_{gen} deteriorates, *PosNet* provides very promising results, reaching nearly state-only models performances on the 'Ellipsoidal bowl' experiments (11.8 pixel prediction error vs 6.1 of NPE). Fig. 6 shows the error evolution through time. This plot shows that in the long term our predictions seems to degenerate quicker than state-only methods

[2]We did not see the need to use an architecture incorporating a gating mechanism, such as a Conv-LSTM Xingjian et al. (2015), because in our case the transition function $\phi_s(s_t)$ does not observe new evidence after the first T_0 frames rendering the use of gating less useful.

Table 2: **Long term predictions.** All of our models (below thick line) observed the $T_0 = 4$ first frames as input. All networks have been trained to predict the $T_{\text{train}} = 20$ first positions, except for the NPEs which were given $T_0 = 4$ states as input and train to predict state at time $T_0 + 1$. We report here results for time $T_{\text{train}} = 20$ and $T_{\text{gen}} = 40$. Unless noted, reported models are trained to predict position and angular velocity. For each time we report on the left average pixel error and root squared L^2 angular velocity loss on the right. Perplexity ($\log_e$ values shown in the table) is defined as $2^{-\mathbb{E}[\log_2(p(x))]}$ where p is the estimated posterior distribution. This value is shown in bracket.

Method	State	Hemispherical bowl Errors (Perplexity)				Ellipsoidal bowl Errors (Perplexity)				Heightfield Errors (Perplexity)			
		T_{train}		T_{gen}		T_{train}		T_{gen}		T_{train}		T_{gen}	
		pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.
Linear	GT	39.2	7.5	127.5	17.9	61.9	23.3	20.1	80.0	21.3	9.4	61.9	19.3
Quadratic	GT	164.3	18.4	120.1	861.2	11.7	14.8	93.1	70.6	26.7	27.4	126.0	122.2
NPE w/o ang. vel.	GT	2.6	–	6.0	–	3.2	–	**6.1**	–	12.0	–	38.5	–
NPE	GT	3.3	0.8	9.6	1.7	2.7	1.4	7.6	2.9	10.9	3.7	32.9	4.6
V-LSTM w/o ang. vel.	Visual	6.3	–	57.5	–	3.2	–	30.4	–	8.8	–	26.7	–
V-LSTM	Visual	5.7	1.3	35.0	2.5	4.0	0.8	39.9	6.0	8.8	2.2	26.1	2.9
IFS w/o ang. vel.	GT	1.3	–	2.9	–	3.3	–	8.9	–	10.4	–	27.6	–
IFS	GT	1.6	**0.3**	**2.2**	**0.4**	3.1	1.0	6.9	**1.4**	8.7	2.5	26.1	2.8
VIN w/o ang. vel.	Visual	40.4	–	37.8	–	24.0	–	30.2	–	42.6	–	42.7	–
PosNet w/o ang. vel.	Visual	**1.0**	–	18.1	–	1.6	–	24.4	–	7.2	–	24.6	–
PosNet	Visual	**1.0**	0.4	13.8	3.0	**1.2**	**0.5**	11.8	3.0	6.8	2.1	23.2	4.2
DispNet w/o ang. vel.	Visual	3.0	–	29.7	–	2.5	–	20.6	–	7.7	–	25.8	–
DispNet	Visual	3.5	1.2	15.9	4.3	**2.1**	1.0	16.1	4.4	7.2	**2.0**	**21.6**	3.3
ProbNet w/o ang. vel.	Visual	2.9 **(4.5)**	–	24.2 (21.9)	–	2.9 (32.1)	–	21.8 (54.0)	–	**6.4** **(9.5)**	–	22.5 (12.7)	–
ProbNet	Visual	3.4 (4.7)	1.2	15.3 **(9.2)**	3.4	4.0 **(4.5)**	1.8	16.7 **(9.3)**	3.8	6.8 (10.8)	2.1	20.5 **(12.3)**	**2.7**

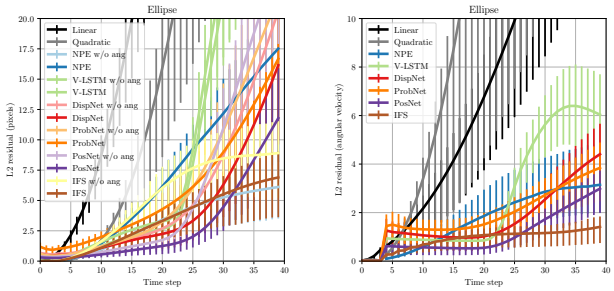


Fig. 6: **Errors evolution on Ellipsoidal Bowl** Position errors (left) and angular velocity error (right). We see that position and angular velocity errors degenerate outside training regimes (t=20) for all non state-only methods with an effect more tempered for our method. The impact is more moderate on angular velocity since its range is smaller than positions. Error bar shows 25th and 75th percentiles.

outside training regimes but still remains more moderate than the other baselines.

We also note that learning to regress *angular velocity* generally improve the ability of our models to predict *position*, in particular when generalizing to T_{gen} steps. For example, PosNet in the Ellipsoidal bowl reduces its position error from 24.4 to 11.8 at T_{gen} when it is required to predict angular velocity during training. For further comparisons, see the similarly colored, adjacent bars in Fig. 5 (left) and Fig. 8 (left)). This is remarkable as angular velocity as such remains very challenging to predict.

An interesting question is whether the model learns or not to measure angular velocity from images, or whether predicting this quantity during simply induces a better internal understanding of physics. To tease this effect out, we prevent the network from observing the ball spin by removing the texture on the ball. Table 3 shows that this results approximately in the same accuracy as the textured cases, indicating that angular velocity is not estimated visually. Our hypothesis is that angular velocity is estimated by exploiting the strong correlation between linear and angular velocities due to conservation of momentum.

Finally, introducing the probability-based loss in DispNet results in the ProbNet network. As shown in Table 2, This change significantly outperforms the deterministic DispNet results in most cases.

(ii) Experiments using multiple balls. We also trained our models with two and three balls in the 'Ellipsoidal bowl' environment to study the ability of our models to handle object interactions without explicit collision modules. The aforementioned training setups are maintained in these experiments. Quantitatively, Table 4 shows that our models were able to get competitive results *w. r. t.* state-only methods containing explicit collision modules, *e.g.*, NPE. Probabilistic model shows an increase in uncertainty at T_{train}, which reveals that the task to solve were harder due to the chaotic nature of the system. In addition, angular velocity seems to be very challenging to estimate in this case. Qualitatively, Section 5.2.1 shows that collisions are well handled by our model despite not being explicitly encoded.

(iii) Ablation study. To better assess the performance of our model with and without the summation module of Fig. 2 we conducted an ablation study. We trained the *DispNet* network on both the Ellipsoidal bowl 2 balls and the Ellipsoidal bowl 3

Table 3: **Impact of ball texturing on prediction.** We compare the impact of ball texturing on predictions. Table layout and measures are same as Table 2. Results show that ball texture is rather ignored to make predictions.

Method	Ellipsoidal bowl Errors (Perplexity)				Ellipsoidal bowl (no ball texture) Errors (Perplexity)			
	T_{train}		T_{gen}		T_{train}		T_{gen}	
	pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.	pixel	ang. vel.
PosNet w/o ang. vel.	1.6	–	24.4	–	1.6	–	23.7	–
PosNet	**1.2**	**0.5**	**11.8**	**3.0**	**1.1**	**0.6**	**12.7**	**3.5**
DispNet w/o ang. vel.	2.5	–	20.6	–	1.7	–	26.3	–
DispNet	2.1	1.0	16.1	4.4	1.6	1.0	16.2	3.8
ProbNet w/o ang. vel.	2.9 (32.1)	–	21.8 (54.0)	–	3.1 (5.0)	–	24.0 (12.7)	–
ProbNet	4.0 **(4.5)**	1.8	16.7 **(9.3)**	3.8	4.3 **(4.5)**	1.3	15.0 **(8.2)**	**3.5**

Table 4: **Multiple balls experiment.** We extend the 'Ellipsoidal bowl' setup adding more balls. We show that in this case our networks get comparable performances to state-only methods. Table layout and measures are the same as Table 2 except that $T_{train} = 15$ and $T_{\text{gen}} = 30$.

Method	States	Ellipsoidal bowl 2 balls Errors (Perplexity) T_{train} pixel	ang. vel.	T_{gen} pixel	ang. vel.	Ellipsoidal bowl 3 balls Errors (Perplexity) T_{train} pixel	ang. vel.	T_{gen} pixel	ang. vel.
NPE	GT	5.3	1.5	13.4	2.0	5.0	1.6	13.3	2.0
V-LSTM	Visual	5.5	2.5	24.1	3.6	6.6	3.9	22.1	4.5
IFS	GT	4.1	**1.3**	**9.6**	**1.5**	**4.3**	**1.5**	**10.0**	**1.6**
PosNet	Visual	4.2	2.4	11.7	2.8	5.7	4.0	15.6	4.5
DispNet	Visual	**3.6**	2.2	16.8	4.1	5.1	3.7	15.9	4.9
ProbNet	Visual	5.3 (7.0)	2.5	19.8 (14.0)	3.6	6.5 (7.5)	3.9	17.1 (12.6)	4.1

balls dataset with and without the extension (with an ablation of the yellow part in Fig. 2). We first evaluated its performance on long-term predictions. Then we studied how both of these models are handling collisions.

Table 5: **Effect of multiple-ball module on extrapolation.** We study the impact of our multiple objects module of Fig. 2on the quality of extrapolation in a multiple-ball scenario. We report results for *DispNet* trained with and without the module on long-term prediction tasks. Table layout and measures are the same as Table 2 except that $T_{train} = 15$ and $T_{\text{gen}} = 30$.

Fig. 2 Module	Ellipsoidal bowl 2 balls Errors T_{train} pix.	ang. vel.	T_{gen} pix.	ang. vel.	Ellipsoidal bowl 3 balls Errors T_{train} pix.	ang. vel.	T_{gen} pix.	ang. vel.
×	**3.6**	**2.2**	17.6	**3.7**	5.2	3.9	18.2	**4.8**
✓	**3.6**	**2.2**	**16.8**	4.1	**5.1**	**3.7**	**15.9**	4.9

We show in Table 5 that the model performs similarly on training regimes (same errors for two balls, 5.2/5.1 and 3.9/3.7 for 3 balls). The module seems to have a clear advantage on long-term predictions where pixel errors are respectively 17.6/16.8 for two balls and 18.2/15.9 for three balls. Angular velocity errors is marginally better without the module, however, both errors remain very close (3.7/4.1 and 4.8/4.9).

Furthermore, we study the impact of our module on collisions. To this end, we created a new dataset extracted from the test data of the multiple balls experiment. In this dataset, we run a collision detector and clipped the experiment at 10 time steps prior to the first observed collision between the balls. In Table 6 we report error numbers at $T = 5$ and $T = 10$ after the collision. We see that our module enables our pipeline to better handle collisions between objects.

Table 6: **Effect of multiple-ball module on collision estimation.** We study the impact of our multiple objects module of Fig. 2 on collision estimation. All experiments start at $T_0 = T_{\text{first collision}} - 10$ for the two multiple balls dataset. We report results for *DispNet* trained with and without the module trained on the extrapolation task in section 5.2.1 with $T_{train} = 15$ and $T_{\text{gen}} = 30$. Table layout and measures are the same as Table 2. We report error at different time T after collision occur.

Fig. 2 Module	Ellipsoidal bowl 2 balls Errors $T = 5$ pix.	ang. vel.	$T = 10$ pix.	ang. vel.	Ellipsoidal bowl 3 balls Errors $T = 5$ pix.	ang. vel.	$T = 10$ pix.	ang. vel.
×	3.3	2.9	6.6	2.7	4.5	4.1	8.3	3.9
✓	**2.6**	**2.3**	**5.3**	**2.5**	**3.9**	**3.5**	**6.7**	**3.7**

(iv) Real data. We also investigate extrapolation on real data using the Roll4Real dataset by Ehrhardt et al. (2018). In our setting, we are only interested in using their unsupervised signal as ground truth position to train our models. We do not address the complex problem of obtaining this signal from unsupervised data. We report the results in Table 7, where all models were trained to predict position only. In each scenario, our models were able to handle the transition to real data as opposed to the baselines. For instance for an ellipsoidal bowl (Ellipsoidal bowl dataset in Table 2 and BowlR1b in Table 7), errors at T_{train} for models trained without angular velocity, went from 1.6 in Table 2 to 5.6 in Table 7 for *PosNet* whereas the error went from 3.3 to 26.2 for IFS. The errors at T_{gen} in this case being generally large (> 23) for models trained without angular velocity.

Table 7: **Long term predictions using real data.** All models are trained using the unsupervised tracker output of Ehrhardt et al. (2018), with the same name for every dataset. Reported number are **pixel errors** for every time. State are the same as Table 2. First three dataset use one ball while last one uses two balls. In all experiment $T_{\text{gen}}=2\times T_{\text{train}}$.

Method	PoolR1b pix. err, $T_{train} = 15$ T_{train}	T_{gen}	HeightR1b pix. err, $T_{train} = 20$ T_{train}	T_{gen}	BowlR1b pix. err, $T_{train} = 20$ T_{train}	T_{gen}	HeightR2b pix. err, $T_{train} = 15$ T_{train}	T_{gen}
V-LSTM	6.5	30.4	6.1	31.3	10.9	58.8	19.0	38.2
IFS	26.0	37.5	48.0	58.1	26.2	39.1	15.6	26.6
VIN	50.9	40.8	40.2	47.3	33.9	33.0	45.9	39.8
PosNet	4.6	21.4	**5.6**	29.0	**5.6**	23.0	**5.4**	**12.5**
DispNet	**3.8**	23.6	**5.6**	**28.5**	6.5	**22.6**	6.2	15.4
ProbNet	4.7(6.)	**16.3**(11.)	5.7(6.)	30.0(22.)	6.8(7.)	23.5(14.)	6.8(8.)	16.9(12.)

5.2.2. Interpolation

So far, we have consider the problem of extrapolating trajectories without any information on the possible final state of the system. We aim here to study the impact of injecting such knowledge in our networks.

In order to do so, in this experiment we concatenate to the first $T_0 = 4$ input frames the last observed frame at time T_{final} and give the resulting stack as input to the encoder network $h_0 = \phi_{\text{enc}}(\mathbf{x}_{(-T_0,0]}, x_{T_{\text{final}}})$ to estimate the first state h_0. In this setting, the model performs "interpolation" as it sees images at the beginning as well as the end of the sequence. The rest of the model works as before with the exception that the first state h_0 is decoded in a prediction $(y_0, y_{T_{\text{final}}}) = \phi_{\text{dec}}(h_0)$ of both the first and the last position $y_{T_{\text{final}}}$; in this manner, the loss encourages state h_0 to encode information about the last observed frame $x_{T_{\text{final}}}$.

Table 8 indicates that the ability of observing an image of the final state enables our models to provide far better estimations. Even in the more complex scenarios with 2 and 3 balls and the heightfield experiments, the errors are significantly lower than for extrapolation. As expected, for InterpNet the highest errors are always found in the middle of the estimate as these points are less predictable from the available information; by contrast, for DispNet the highest errors are at the end.

Still, we note that harder scenarios result in larger errors even for interpolation, and particularly for colliding balls due to the chaotic nature of this dynamics. This also shows the current limitation of our system in modeling collisions and complex variable environments.

5.3. Discussion

In addition to the various results we presented, we discuss our conclusions regarding the main sources of prediction error in the conducted experiments.

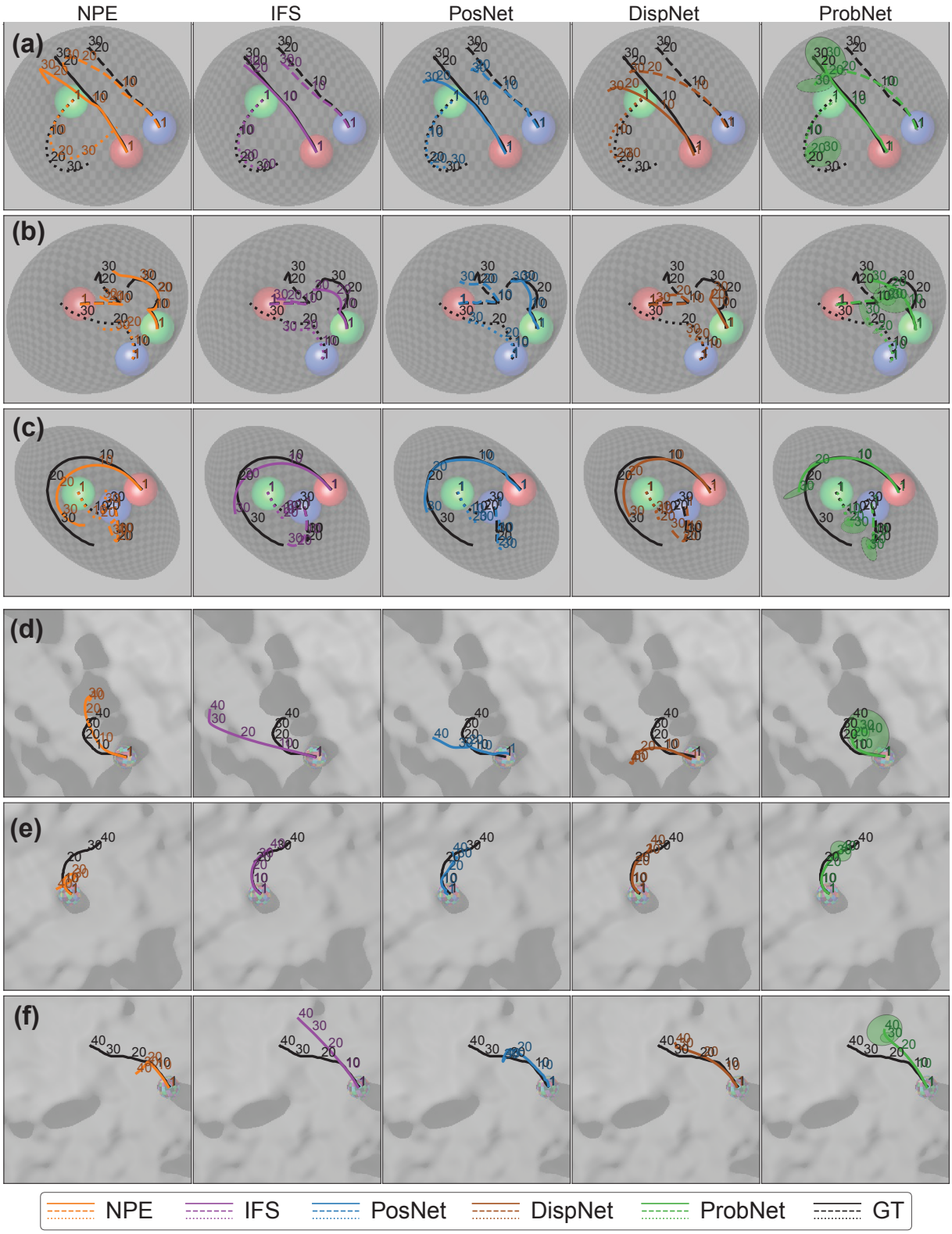


Fig. 7: **Ellipsoidal bowl and Heightfield extrapolations.** (a-c) Example scene from the 3 balls in the 'Ellipsoidal bowl' experiment. Extrapolation on multiple objects generalises well to 3 objects. Note how in (b) the collision of the red and green ball is predicted by our networks, solely by seeing the first 4 frames of the sequence. Remember, NPE and IFS start with the ground truth knowledge of the physical state of the objects. (d-f) Our models, taking only 4 images as input, have learned to parse the illumination of a quickly changing heightfield surface and use it to predict the long-term (up to 10x the length of initial observation) motion of an object. (d) For homogeneously lit flat regions, it is difficult to make decisions, indicated by *ProbNet*'s large uncertainty estimates. (e) IFS, *DispNet* and *PosNet* correctly interpret the ball's initial angular velocity to predict the future path. *ProbNet* demonstrates the power of anisotropic uncertainty estimation (c, f). It is more certain in the direction of motion than orthogonal to it. Note, that NPE and IFS were given the ground truth object positions for the first four frames, and do not have the capability to take images as input.

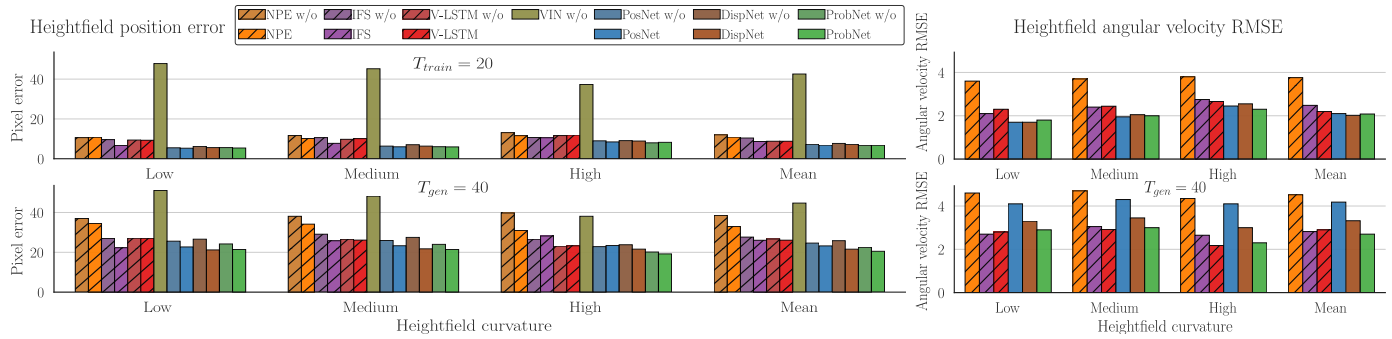


Fig. 8: **Errors on Heightfields.** Position errors (left) and angular velocity error (right) for trained (T_{train}= 20) and untrained (T_{gen}= 40) generalization on increasing difficulty heightfields ('Mean' is reported in the right column of Table 2). Note, how angular velocity estimation helps position accuracy. Hatches denote non-visual methods.

Table 8: **Extrapolation vs Interpolation.** We constructed *InterpNet* as an extension of *DispNet*, where in addition to the concatenation of the first $T_0 = 4$ frames, also the last frame at T_{final} is provided to the model as inputs. All networks have been trained to predict the $T_{\text{train}} := T_{final}$ positions. As expected, *InterpNet* learned to predict the positions at T_{final} by relying on the features extracted from the last input image. We report the **pixel errors** at different times along the sequences. T_{final} is the last value shown for every experiment.

Method	Hemispherical bowl pixel error, $T_{\text{train}} = 40$				Ellipsoidal bowl 1 ball pixel error, $T_{\text{train}} = 40$				Ellipsoidal bowl 2 balls pixel error, $T_{\text{train}} = 30$			Ellipsoidal bowl 3 balls pixel error, $T_{\text{train}} = 30$			Heightfield pixel error, $T_{\text{train}} = 40$			
	T=10	20	30	40	T=10	20	30	40	T=10	20	30	T=10	20	30	T=10	20	30	40
DispNet	2.2	3.6	3.9	5.0	1.4	2.4	2.7	3.0	2.8	5.8	8.7	3.2	8.1	12.0	3.6	7.9	12.9	17.9
InterpNet	1.4	1.8	1.6	1.0	1.0	1.6	1.3	0.6	3.2	4.5	3.1	3.3	4.5	2.1	2.5	5.2	5.1	1.6

Does training for longer horizons help? Training for longer horizons T_{train}= 40 in Table 8 compared to T_{train}= 20 in Table 2 results in better position estimates as expected. When a single end state is also observed (interpolation) the model manages to infer plausible trajectories even though the initial and final states are far apart in time.

Table 9: **Length of supervision.** The **maximum** position error of *DispNet* decreases when we add more supervision during training.

Dataset	Extrapolation		Interpolation
	T_{train}= 20	T_{train}= 40	T_{train}= 40
'Hemispherical bowl'	15.9	5.0	1.8
'Ellipsoidal bowl'	16.1	3.0	1.6
'Heightfield'	21.6	17.9	5.2

This motivates us to design more structured representations in the future, which would generalize even better outside the supervised time spans (see Table 9).

Can the models handle collisions of multiple objects? Adding additional objects to our scenes has appeared to be a challenging task for our models. If our multiple objects module helped to better handle collisions (see Table 6), the error increased with the number of objects, which shows that collisions remain difficult to estimate. Promisingly, *InterpNet* manages to improve performance similarly to the earlier cases, the remaining ambiguity in the middle of the sequences matches the ratios of single object examples ($Error_{T=10}/Error_{T=20}$: $1.0/1.6 \simeq 3.2/4.5 \simeq 3.3/4.5$ in Table 8 middle columns).

Does regression of angular velocity help? Almost all models benefit from the additional supervision signal coming from the loss on angular velocity, as shown in Fig. 5(left) and Fig. 8 (left). The objects' texture at these resolutions is difficult to interpret, and the connection between pixel color and rotation around axis is highly non-linear, which encourages us to look for a different representation of rotation in the future to improve our angular prediction errors.

Are changing environments more difficult? The characteristics of the environment also appear to strongly contribute to the final estimation errors. When only following one ball we notice that for simple shapes where the environment parameters can vary along at most 3 dimensions (in the 'Hemispherical bowl' and 'Ellipsoidal bowl' cases), the system can obtain nearly perfect estimates in the interpolation experiments. However in the 'Heightfield' scenes interaction with the environment is much more difficult to estimate and the maximum errors are larger, even for *InterpNet* the errors remains substantial.

6. Conclusions

In this paper, we studied the possibility of abstracting knowledge of physics using a single neural network with a recurrent architecture to model long term predictions with a changing environment. We compared our model to various baselines on the non-trivial motion of ball(s) rolling on a surfaces with different possible shapes (*e.g.* ellipsoidal bowls or randomized heightfields) on both synthetic and real data. Closer to some approaches, we do *not* integrate physical quantities but implicitly encode the states in a feature vector that we can propagate through time.

However, we demonstrated a significant difference compared to existing networks using implicit state encoding, namely the ability to account for complex variable environments. The latter

leverage a distributed representation of the system state which, at the same time, is still able to model concentrated object interactions such as collisions.

Our experiments on synthetic simulations also indicate that our networks can predict mechanical phenomena more accurately than networks that build on hand-crafted physically-grounded representations of the system state. This means that our approach can both infer automatically an internal representation of these phenomena and work with visual inputs in order to initialize such a representation and use it for extrapolation. Our models can also estimate a distribution over physical measurements such as position to account for uncertainty in the predictions.

While keeping the same architecture, we further demonstrate that it is possible to remove ambiguity by showing the network an image of the final state of the system, performing interpolation. However, in this case the internal state propagation mechanism is still limited by its ability to make accurate long term predictions outside temporal spans observed during training.

In the future, we aim at increasing the robustness and generalization capabilities of our models by enforcing more explicitly temporal and spatial invariance (as physical laws are constant and homogeneous). Finally, we plan to work on the generalization abilities of our multiple objects pipeline to handle various object shapes and remove the limitation of having to known the number of objects in advance.

Acknowledgments

The authors would like to gratefully acknowledge the support of ERC 638009-IDIU and ERC SmartGeometry StG-2013-335373 grants

References

Abadi, M., et al., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Agrawal, P., et al., 2016. Learning to Poke by Poking: Experiential Learning of Intuitive Physics, in: Proc. NeurIPS, pp. 5074–5082.

Battaglia, P., et al., 2016. Interaction networks for learning about objects, relations and physics, in: Proc. NeurIPS, pp. 4502–4510.

Battaglia, P.W., Hamrick, J.B., Tenenbaum, J.B., 2013. Simulation as an engine of physical scene understanding. PNAS 110, 18327–18332.

Chang, M., et al., 2017. A compositional object-based approach to learning physical dynamics, in: Proc. ICLR.

Chiappa, S., et al., 2017. Recurrent environment simulators., in: Proc. ICLR.

David, S., et al., 2016. The predictron: End-to-end learning and planning. CoRR abs/1612.08810. URL: `http://arxiv.org/abs/1612.08810`.

De Brabandere, B., et al., 2016. Dynamic filter networks, in: Proc. NeurIPS.

Denil, M., et al., 2016. Learning to perform physics experiments via deep reinforcement learning. Deep Reinforcement Learning Workshop, NIPS .

Ehrhardt, S., et al., 2017a. Learning A Physical Long-term Predictor. arXiv e-prints arXiv:1703.00247 `arXiv:1703.00247`.

Ehrhardt, S., et al., 2017b. Learning to Represent Mechanics via Long-term Extrapolation and Interpolation. arXiv preprint arXiv:1706.02179 `arXiv:1706.02179`.

Ehrhardt, S., et al., 2018. Unsupervised Intuitive Physics from Visual Observations. Proc. ACCV .

Fragkiadaki, K., et al., 2016. Learning visual predictive models of physics for playing billiards .

Greff, K., et al., 2017. Neural expectation maximization, in: Proc. NeurIPS.

Hamrick, J.B., et al., 2016. Inferring mass in complex scenes by mental simulation. Cognition 157, 61–76.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780.

Krizhevsky, A., Sutskever, I., Hinton, G., 2012. Imagenet classification with deep convolutional neural networks, in: Proc. NeurIPS, pp. 1097–1105.

Ladický, et al., 2015. Data-driven fluid simulations using regression forests. ACM Trans. on Graphics (TOG) 34, 199.

Lerer, A., Gross, S., Fergus, R., 2016. Learning physical intuition of block towers by example, in: Proc. ICML, pp. 430–438.

Li, W., Leonardis, A., Fritz, M., 2017. Visual stability prediction and its application to manipulation. AAAI .

Monszpart, A., Thuerey, N., Mitra, N., 2016. SMASH: Physics-guided Reconstruction of Collisions from Videos. ACM Trans. on Graphics (TOG) .

Mottaghi, R., et al., 2016. Newtonian scene understanding: Unfolding the dynamics of objects in static images, in: IEEE CVPR.

Oh, J., et al., 2015. Action-conditional video prediction using deep networks in atari games, in: Proc. NeurIPS, pp. 2863–2871.

Ondruska, P., Posner, I., 2016. Deep tracking: Seeing beyond seeing using recurrent neural networks, in: Proc. AAAI.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition, in: Proc. ICLR.

Stewart, R., Ermon, S., 2017. Label-free supervision of neural networks with physics and domain knowledge, in: AAAI, pp. 2576–2582.

Tieleman, T., Hinton, G., 2012. Lecture 6.5—RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Tompson, J., et al., 2016. Accelerating Eulerian Fluid Simulation With Convolutional Networks. ArXiv e-print arXiv:1607.03597 .

Villegas, R., et al., 2017. Learning to generate long-term future via hierarchical prediction., in: Proc. ICML.

Watters, N., et al., 2017. Visual interaction networks: Learning a physics simulator from video, in: Proc. NeurIPS. Curran Associates, Inc., pp. 4542–4550.

Wu, J., et al., 2015. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning, in: Proc. NeurIPS, pp. 127–135.

Wu, J., et al., 2016. Physics 101: Learning physical object properties from unlabeled videos, in: Proc. BMVC.

Wu, J., et al., 2017. Learning to see physics via visual de-animation, in: Proc. NeurIPS.

Xingjian, S., et al., 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting, in: Proc. NeurIPS, pp. 802–810.

Xue, T., et al., 2016. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks, in: Proc. NeurIPS.

Yu, K., et al., 2016. More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing, in: Proc. IROS, IEEE. pp. 30–37.

Chapter 4

Unsupervised intuitive physics from visual observations

While chapter 3 proposed a method to extrapolate motions of objects from synthetic and real sequences, it still requires labelling of the sequences. This becomes an issue for real data since labelling sequences of hundreds of frames becomes manually infeasible when the number of sequences becomes large. In addition, when using weakly-supervised trackers from past works we found that their performances are not always reliable in particular when there are small jitters in a scene.

In this chapter, we propose a self-supervised method designed to track moving objects in scenarios close to the ones introduced in chapter 3. This tracker is based on three key properties: *motion continuity, low entropy and rotation equivariance.* We also contribute to a new dataset of real data: Roll4Real. We show that our method has robust performances for our proposed dataset, superior to current best trackers. Then, the output of this tracker is used to produce a supervision signal which is in turn used to make an extensive benchmark of long term physical prediction methods on real data.

This work was presented at the Asian Conference of Computer Vision, 2018.

Unsupervised Intuitive Physics from Visual Observations

Sebastien Ehrhardt[1⋆][0000−0002−6948−079X], Aron Monszpart[2,3⋆][0000−0003−0579−5300], Niloy Mitra[2][0000−0002−2597−0914], and Andrea Vedaldi[1][0000−0003−1374−2858]

[1] University of Oxford, Oxford, UK, {hyenal, vedaldi}@robots.ox.ac.uk
[2] University College London, London, UK, {a.monszpart,n.mitra}@cs.ucl.ac.uk
[3] Niantic, San Fransisco, USA, aron@nianticlabs.com

Abstract. While learning models of *intuitive physics* is an active area of research, current approaches fall short of natural intelligences in one important regard: they require external supervision, such as explicit access to physical states, at training and sometimes even at test time. Some approaches sidestep these requirements by building models on top of handcrafted physical simulators. In both cases, however, methods cannot learn automatically new physical environments and their laws as humans do. In this work, we successfully demonstrate, for the first time, learning unsupervised predictors of physical states, such as the position of objects in an environment, *directly from raw visual observations and without relying on simulators*. We do so in two steps: (i) we learn to track dynamically-salient objects in videos using causality and equivariance, two non-generative unsupervised learning principles that do not require manual or external supervision. (ii) we demonstrate that the extracted positions are sufficient to successfully train visual motion predictors that can take the underlying environment into account. We validate our predictors on synthetic datasets; then, we introduce a new dataset, Roll4Real, consisting of real objects rolling on complex terrains (pool table, elliptical bowl, and random height-field). We show that it is possible to learn reliable object trajectory extrapolators from raw videos alone, without any external supervision and with no more prior knowledge than the choice of a convolutional neural network architecture.

Keywords: Unsupervised learning · Motion · Convolution Networks.

1 Introduction

A striking property of natural intelligences is their ability to perform accurate and rapid predictions of physical phenomena using only noisy sensory inputs. Even more remarkable is the fact that such predictors are learned without explicit supervision; rather, natural intelligences induce their internal representation of physics automatically from experience.

⋆ Authors contributed equally

Several authors have recently looked into the problem of learning physical predictors using deep neural networks in order to partially mimic this functionality. Early attempts predicted trajectories in hand-crafted spaces of physical parameters, such as positions and velocities, assuming that the ground-truth values of such parameters are fully observable during training. Others have considered performing predictions from visual observations, but used full supervision for training. Furthermore, while several papers [7,3] make use of simulators as a way to generate the required supervisory signals, limited work has been done in transferring such models to real data.

In this paper, we also investigate learning *physical predictors* using deep neural network. However, we do so in a **fully unsupervised manner**, learning from observations of unlabelled video sequences. In contrast to approaches such as the recent de-animation method of [39], we do not require synthetic data, nor do we rely on any handcrafted physical simulator for prediction. Our models are built directly from real data and learn intuitive physics models that empirically outperform more principled, but more brittle, models based on physical parameters [31].

Importantly, our goal is not to merely predict future frames in a video, a problem addressed before by several authors [19]. While we also predict future dynamics from a video stream, our goal is *not* to estimate appearance changes, but physical quantities such as object positions and velocities. So, where future frame prediction generates an image, our goal is to extract meaningful and actionable physical parameters from the data.

As a working example, we consider video footage of balls rolling on various surfaces, such as pool tables, bowls and random height-fields. Balls interact with the underlying environment (e.g., roll around obstacles) and among themselves (e.g., collide with each other). For rigorous assessment, in addition to considering several synthetic datasets, we also contribute **a new public dataset**, Roll4Real, containing a large number of such sequences captured in real-life. Methodologically, we make two contributions. First, inspired by [23], we show that an object **detector** can be learned in an **unsupervised manner** by tuning a convolutional detector to extract tracks that are maximally characteristic of the natural, causal ordering of the frames in a video. Second, we use these trajectories to learn **visual predictors** that automatically learn an internal representation of physics and can extrapolate the trajectory of the balls more reliably than even supervised approaches such as Interaction Networks (IN) [3] that use direct measurements of physical parameters.

Note that our goal, similar to other papers in this area, is not to come up with the best possible method for physical prediction. A handcrafted solution heavily engineered to use supervision, off-the-shelf trackers, and/or physical simulators may do better in raw predictive performance (although the task is in fact not simple, particularly as our terrains are complex and somewhat deformable). Rather, we focus on developing machines that can learn such physical predictors from raw input.

Empirically, we show that vision-based models more gracefully handle observation noise compared to approaches such as [7,3] that are learned using physical ground-truth parameters extracted from simulated scenarios. We also show that the Visual Interaction Network (VIN) of [22], which also propose a vision-based physical predictor, fails to account for the interaction of the objects and their environment, whereas more distributed tensor based approach succeeds.

The rest of the paper is organized as follows. We discuss related work in section 2. We then present the technical details of our approach in section 3. Next, we introduce the new Roll4Real data in section 4 and use the latter as well as several existing synthetic datasets to evaluate the approach in section 5. We summarise our findings in section 6.

2 Related Work

Existing work in learning physics can be organised according to several axes.

Nature of the Representation of Physics: A natural way to represent physics is to manually encode every object parameters and physical properties (mass, velocity, positions, etc.). From the earliest approaches [4] this has been widely used to represent physics and propagate it [3,7,26,32]. Some focusing on representing a small subset of physical parameters such as positions and velocities [37,38]. However, other approaches try to learn an implicit representation of physics, inspired by the success of implicit representation of dynamics [29,28,8,5]. Implicit physics are usually represented as activations in a deep neural network [10,36,20].

Hand-crafted vs Learned Dynamics: Some approaches [37], including simulation-based ones [4,40], use physics by explicitly integrating parameters such as velocities. While this generally require extensive knowledge of the environment and object properties, other methods [3,7,26,32,10], integrate parameters of the scenarios through recurrent learnable predictors to make physical long term predictions.

Physical vs Visual Observations: Many approaches [3,7] assume direct access to physical quantities such as positions and velocities for prediction. If this first approach enable to make very accurate predictions it is however unlikely that such accuracy can be reached in the real-world. Others [4,20,21,40,36,13,33,17] take as input one or several frames of a scene to deduce physical properties (intuitive or explicit) or predict the next state of a system.

Qualitative vs Quantitative Predictions: While most of the papers discussed above consider *quantitative* predictions such as extrapolating trajectory, others have considered *qualitative* predictions focusing on *intuitive* physics, such as the stability of stacks of objects [4,20,21], the likelihood of a scenario [30] or the forces acting behind a scene [40]. Other papers are in between, and learn *plausible if not accurate physical predictions* [35,18,24], often for 3D computer graphics.

Nature of the Supervision: Most approaches are *passive and supervised*, as they are passive observer of physical scenarios and use ground truth information

about key physical parameters (positions, velocities, stability) during training. While this approaches require an expensive annotation of data, some work tried to learn from unsupervised data either through active manipulation [2,9] or using the laws of physics [33].

Scenarios: Two favorite scenarios in such experiments are bouncing balls, including billard-like environment [14], and block towers [20]. As a variant, [36] consider balls subject to gravitational pulls, ignoring harder-to-model collisions. Most papers make use of simulated data, with limited validation on real data. A different approach [25] is to predict qualitative object forces and trajectories in fully-unconstrained natural images. The approach of [2] considers instead learning from active poking using a real-life robot. In most cases experiments are done on synthetic data. However, approaches such as [37,38,21] also used real data; [38] also contributed a dataset of videos of short physical experiments called *Phys-101*.

We relate to such previous work in that we also make physical predictions of the trajectory of ball-like objects. However, we differ in two significant ways. First, our approach, while using only passive observations, is *fully unsupervised*, and yet competitive if not more accurate than supervised counterparts. In particular, while [33,40] also do not use image labels, they use *a-priori* knowledge of physics for training (a fully-fledged simulator and renderer in the case of [40]). Second, we systematically test on *several real-life scenarios*, both in training and testing, using our new ROLL4REAL dataset. Compared to datasets such as *Phys-101*, ours allows testing long-term ball-rolling prediction in complex scenarios.

3 Method

Our goal is to construct a machine that can, given only raw videos and no supervision, learn physical parameters such as the position of the objects in the videos as well as proxies to physical laws that allow to predict the evolution of such parameters over time. For this, predicting appearance changes is not sufficient; instead, we decompose the problem in two steps. The first one is a method to discover and learn to extract object positions using as cue the fact that they should have a non-trivial causal dynamics (section 3.1). This tracker scales well to large datasets and is able to detect different type of objects without any further specification. Then, we use the resulting object trajectories to learn visual predictors that can extrapolate the object positions through time, embodying a proxy to the laws of mechanics (section 3.2).

3.1 Unsupervised Detection and Tracking of Dynamic Objects

Single-object Detection. Let $\mathbf{x}_t \in \mathbb{R}^{H\times W\times 3}$ be a RGB video frame and assume we are given video sequences $\mathcal{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, initially containing a single object moving in an environment, such as a rolling ball. Our goal is to learn a detector function $\Phi(\mathbf{x}_t) = u_t \in \mathbb{R}^2$ that extracts the 2D position u_t of the

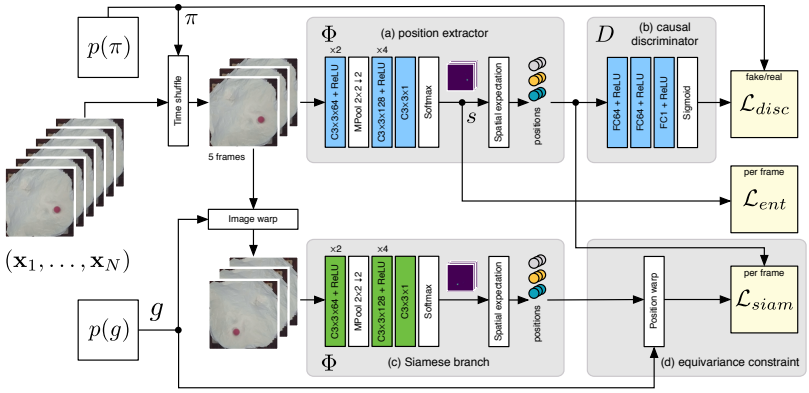


Fig. 1. Overview of our unsupervised object tracker Each training point consists of a sequence of five video frames. Top: the sequence is randomly permuted with probability 50%. The position extractor (a) computes a probability map s for the object location, whose entropy is penalised by $\mathcal{L}_{ent}$. The reconstructed trajectory is then fed to a causal/non-causal discriminator network (b) that determines whether the sequence is causal or not, encouraged by $\mathcal{L}_{disc}$. The bottom Siamese branch (c) of the architecture takes a randomly warped version of the video and is expected by $\mathcal{L}_{siam}$ to extract correspondingly-warped positions in (d). Blue and green blocks contain learnable weights and green blocks are siamese shared ones. At test time only Φ is retained.

moving object at any given time (Fig. 1). The challenge is to do so *without* access to any label for supervision or any a-priori information about object shape.

We start by implementing $\Phi(\mathbf{x}_t)$ as a shallow Convolutional Neural Network (CNN) that extracts a scalar score $f_v \in \mathbb{R}$ for each image pixel $v \in \Omega = \{1, \ldots, H\} \times \{1, \ldots, W\}$, resulting in a heat map. This is then normalised to a probability distribution using the softmax operator $s_v = e^{f_v} / \sum_{z \in \Omega} e^{f_z}$ and the location u of the object is obtained as the expected value $u = \sum_v v s_v$ [13].

We learn Φ by combining two learning principles. The first one is **causality**. Applied to a video sequence, the detector produces a trajectory $\Phi(\mathcal{X}) = (\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_N))$. We expect that, when the detector locks properly on the rolling object, the trajectory is physically plausible (e.g., causal/smooth); at the same time, if the frames are shuffled by a random permutation π, the resulting trajectory should *not* be plausible anymore. We incorporate this constraint by learning a discriminator network $D(\Phi(\mathbf{x}_{\pi_1}), \ldots, \Phi(\mathbf{x}_{\pi_5}))$ that, for a subsequence, can distinguish between the natural ordering of the frames and a random shuffle (top row of Fig. 1). The permutation π is sampled with 50% probability as a consecutive sequence of 5 frames ($\pi_{i+1} = \pi_i + 1$, $i = 1, \ldots, 4$) and with 50% uniformly at random. The discriminator is a 3 layers multi-layer perceptron followed by a sigmoid and the loss

$$\mathcal{L}_{disc}(D, \pi) = \begin{cases} -\log D, & \pi_{i+1} = \pi_i + 1,\ i = 1, \ldots, 4, \\ -\log(1 - D), & \text{otherwise.} \end{cases}$$

The second learning principle is **equivariance** (cf., [34,27]). This principle suggests that, if a transformation g is applied to a frame $\mathbf{x}_t$ (e.g., a $\pm\pi/2$ rotation), then the output of the detector should change accordingly: $\Phi(g\mathbf{x}_t) = g\Phi(\mathbf{x}_t)$. This is implemented as a Siamese branch (bottom row in Fig. 1) extracting 2D positions $\Phi(g\mathcal{X}) = (\Phi(g\mathbf{x}_1), \dots, \Phi(g\mathbf{x}_N))$ from the rotated frames and comparing them to the rotated 2D positions extracted from the original frames using the L^2 loss: $\mathcal{L}_{siam} = \frac{1}{N}\sum_t \|g^{-1}\Phi(g\mathbf{x}_t) - \Phi(\mathbf{x}_t)\|^2$.

Finally, in order to encourage the softmax operator to produce peaky distributions, we minimise the entropy of the resulting distribution $\mathcal{L}_{ent} = -\sum_{v\in\Omega} s_v \log(s_v)$. The final loss is therefore $\mathcal{L} = \lambda_d\mathcal{L}_{disc} + \lambda_e\mathcal{L}_{ent} + \lambda_s\mathcal{L}_{siam}$. In our experiment, $\lambda_d = 1$, $\lambda_e = 0.01$, and $\lambda_s = 0.001$.

Multi-object Tracking. We now extend the method from detection of single objects to tracking of multiple objects. In order to do so, the network is fine-tuned to videos containing two or more moving objects of different appearance.

Since the network produces only a single pair of coordinates, it can formally estimate the location of a single object in the image. However, when multiple objects are present, the unsupervised learning process could still converge to an undesirable result, such as predicting the center of mass of several objects combined, or randomly jumping between objects over time. The first is discouraged by the entropy loss which prefers sharp heat map. The second is discouraged by the causality loss, as discontinuous trajectories would not look plausibly ordered and consistent.

In practice, our model learns to track consistently a single object selected at random among the visible ones. Once this is done, in the next iteration, a second object is detected by suppressing (setting to zero) a circular region of radius r around the first object location in the activations f_v immediately preceding the softmax operator, and the process is repeated for further object occurrences. Before the suppression we also add a positive bias to the activations f_v in order to consider the previously detected objects as zero probabilities in the new heatmap. Note that we consider the number of objects as given since it is in itself already a very challenging task that is under active research [12].

3.2 Trajectory Extrapolation Networks

We consider existing network modules for physical prediction. While these modules use external supervision in the original papers, here we apply them to the output of the unsupervised tracker of 3.1, hence training such physical extraploators in a *fully unsupervised* manner for the first time.

We experiment in particular with *PosNet*, *DispNet*, and *ProbNet* from [11], configuring them to take as input the first four frames of a sequence and to produce as output the prediction of future object positions. These models learn an implicit representation of physics, which is extrapolated automatically by a recurrent propagation layer and used to extract estimates of the object positions. The difference between the models is that *PosNet* regresses positions from state, while *DispNet* and *ProbNet* regress displacements from state. Furthermore, *ProbNet* produces a probability estimate over trajectories.

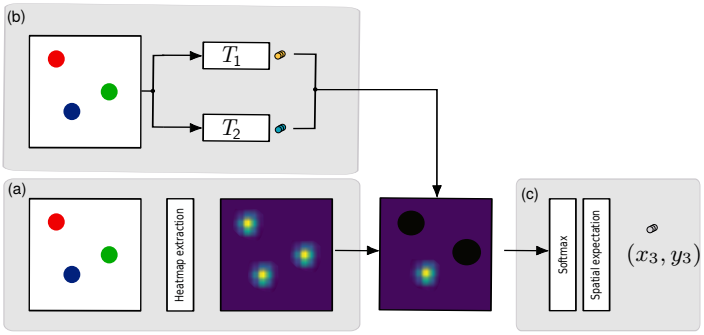


Fig. 2. Multiple object unsupervised tracker (a) We first extract an object heatmap with the method described in 3.1. (b) Then we mask the objects detected by previously trained tracker (T_1 and T_2) on the heatmap by zeroing out the values around a circular area around their center. (c) Finally we extract position from this last heatmap with masked values.

We also consider the *Visual Interaction Network* (VIN) module and its variant *Interaction Network from State* (IFS) [36]. While VIN uses only visual inputs for prediction just like the other networks, IFS works with an explicit state vector of physical parameters, which we set as the stacking of the 2D positions for four past frames which starts with positions extracted from our tracker. Additionally, in the synthetic experiments (first part of Table 2), IFS uses velocity and in BOWLS experiments the ground-truth ellipsoid axes parameters are appended to the state to inform the model of the shape of the ground. IFS and VIN are trained following [11]; in particular, this means that VIN uses the same setting as the original paper (32×32 pixels images).

We also note that while VIN and models from [11] have essentially the same core concepts (they consist of a first feature extractor module to extract implicit physical state, a recurrent propagation module to propagate the state, and an extractor module to get desired physical parameters from the state) their main difference resides in the structure of the propagated state. While VIN used a vector state representation, each of *PosNet*, *DispNet*, and *ProbNet* use a tensor representation.

All such models are trained by showing the network four initial frames of a sequence and the output of the unsupervised tracker up to time $T_{\text{train}} \in \{15, 20\}$ frames. At test time, the networks, which are recurrent, are used to extrapolate the trajectory up to an arbitrary time T, also starting from four video frames. We test in particular $T = T_{\text{train}}$ and $T \gg T_{\text{train}}$ to assess the generalization capabilities of the models learned by the network.

In addition, for some experiments on single object we also consider *linear* and *quadratic* extrapolators as baselines. In both cases we fit a first (respectively second) order polynomial to the 10 first positions given as input (hence with a significant advantage compared to the networks which only observe four frames).

Fig. 3. Physical setup In each of the three real-world scenarios (PoolR, BowlR, HeightR), we show the experimental setup (left) and a sample data frame (right).

4 Roll4Real: A New Benchmark Dataset

In the absence of a suitable real-world dataset to evaluate intuitive physics on objects rolling on complex terrains, we created a new benchmark, Roll4Real (R4R).

Dataset Content. R4R consists of 1118 short 256×256 videos containing one or two balls rolling on three types of terrains (Fig. 3): a flat pool table (PoolR), a large ellipsoidal 'bowl' (BowlR), and an irregular height-field (HeightR). More specifically, there are 151 videos (avg. 99 frames/video) for the PoolR dataset with one ball; 216 videos (522 frames/video) for the BowlR dataset with one ball; 543 videos (356 frames/video) for the HeightR dataset with one ball; and 208 videos (206 frames/video) for the HeightR dataset with two balls. We rolled a total of 7 differently colored balls for the HeightR and BowlR datasets, varying from 3.5 cm to 7 cm in diameter. The height-field surface fits into a $70 \times 70 \times 28$ cm^3 bounding box, with 76 cm diameter. The bowl was created using a 70 cm diameter ball, and is 60 cm high. Videos were randomly split into *train*, *validation*, and *test* sets. Ground-truth annotations are provided for the test split.

Dataset Collection. Both the bowl and height-field terrains were modeled using paper mâché on scaffolds, using a large inflatable ball and a custom-made wire-mesh frame, respectively. For the the PoolR dataset, balls were rolled on the table, while for the other settings, balls were manually dropped from a small height and allowed to roll on. The setup was imaged using a fixed camera (Samsung Galaxy S8) from the top. The PoolR dataset was captured at 30fps (due to low light), while all the others at 240fps in order to reduce motion blur and later downsampled to 80fps. Videos were cropped to only focus on the scenario of interest, i.e., ball(s) and terrain, and trimmed to retain the portion of the video containing motion. We rolled a total of 7 different balls: a pink foam ball (7 cm diameter), a fluorescent yellow tennis ball (6.8 cm), a blue and an orange ping-pong ball (4 cm), a black squash ball wit two yellow dots (4 cm), and a green and a brown cork ball (3.5 cm).

In order to create ground-truth tracks for the ball centers, we used a template-based tracker using zero-normalized cross-correlation in the LAB color space, and tracked each frame along with a smoothness term over time. The setup was manually initiated by providing suitable template. The raw results were then manually inspected, corrected, and saved as ground-truth. We found that due to environment jitter (the ball rolling on the different terrains often created vibration or deformation in the BowlR and HeightR datasets), differences in

lighting across some experiments and different ball colors, the template-based tracker was not perfect and manual inspection was required.

It is worth noting that, while this process was enough to produce ground truth annotations for the test set, the method does not scale due to the need for manual verification and correction. While our aim here is to show the feasibility of learning physics in an unsupervised manner, such problems show that our deep tracker also has an applicative advantage compared to these traditional handcrafted approaches.

5 Results and Discussion

Implementation Details For all networks trained on every dataset, weights were initialised using Xavier initialization [15]. The learning rate was initially set to 10^{-4} and was progressively decreased by a factor of 10 when no improvements were found over K epochs ($K = 100$ for the synthetic datasets). Training was stopped when the loss did not decrease for $2K$ consecutive epochs. Before processing images, we resized all dataset images to 128×128 pixels to fit in the GPU memory. We used TensorFlow [1] on a single NVIDIA Titan X GPU for all the experiments.

5.1 Unsupervised Tracker

We first evaluate our unsupervised object detector and tracker and compare against currently state-of-the-art trackers. We report results in Table 1 against the following trackers: 1. Optical Flow Lucas-Kanade (OFLK) from OpenCV[6] library; 2a. Flownet2-simple, which computes pairwise flowfields using FlowNet2 [16] and follows the velocity vectors; 2b. Flownet2-blob, where we after computing the flowfields from FlowNet2 [16], update the positions as the center of the blobs found in the flowfield. If no blob was detected, we updated the position according to 2a; 3. LAB: a template tracker similar to section 4 without any manual corrections. Note that these methods need manual initialization at the objects positions (expect for LAB) or templating which needs more work with growing object count and/or variety. In addition to PoolR, BowlR, and HeightR from Roll4Real, we also consider two synthetic datasets from [11] in Fig. 4: BowlS for the ellipsoidal bowl with one or two balls and HeightS for the random height-fields. Fig. 4-left reports the mean and 99^{th} percentile pixel error of the extracted object positions against ground-truth averaged over multiple runs of our experiments. Even though the trackers perform well in practice, they suffer from *large variance*. For example, OFLK went off-track 15% of the time on the BowlR dataset, 10% for the HeightR, and 30% for PoolR. In contrast, ours never loses track of the object. The 99th percentile reported in Fig. 4 shows that the offset is almost constant generally due to the detection occurring on the edge of the objects. Overall, our method learns to track objects robustly in a diverse range of complex scenarios.

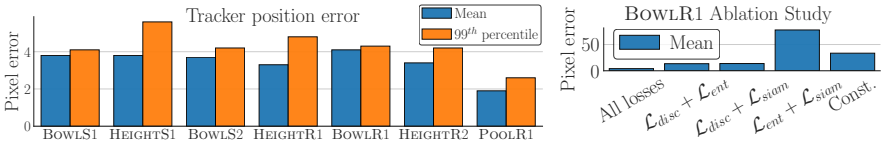


Fig. 4. Tracker errors and Ablation study Left: Tracker errors on different dataset. The errors are consistently small across dataset and show that our tracker can perform well on a different range of real situations. Right: Ablation study. We try different combination of tracker losses on the BowlR dataset. 'Const.' indicates that we are predicting a constant point at the center of the image for reference. For left and right, position errors are reported in pixels. The number of balls in the datasets is appended to the name of the dataset.

Table 1. Tracker results across real datasets The reported numbers are the average (left) and the variance (right) of the pixel error. All numbers refer to 128×128 images.

	PoolR		BowlR		HeightR		HeightR 2B.	
1. Optical Flow Lucas-Kanade	23.3	965	5.6	275	**2.7**	12.9	**2.0**	5.3
2a. FlowNet2-simple	41.4	767	30.4	715	16.6	206	-	-
2b. FlowNet2-blob	3.9	12.1	**2.2**	4.8	4.6	28.7	-	-
3. LAB w/o manual correction	**0.3**	**0.1**	16.4	247	8.3	104	21.7	102
4. Ours	1.9	0.2	4.1	**0.5**	3.3	**0.5**	3.4	**1.2**

Importantly, since our tracker does not use any manual annotations it scales easily to larger synthetic datasets, multiple objects, and different object appearances within the same dataset by just providing more example data.

We also conducted an ablation study on the BowlR dataset to measure the impact of each loss term. Fig. 4-right shows that, while each loss contributes to the final results, the best performance is obtained when all the terms are combined.

5.2 Unsupervised Physics Extrapolation

Supervised vs Unsupervised (Single Ball Synthetic Datasets). We now compare training predictors using either ground-truth object positions or the output of the unsupervised tracker. All predictors observe only $T_0 = 4$ frames as input (either positions or video frames) except VIN which uses $T_0 = 6$ and the least squares baselines which use $T_0 = 10$. All the networks were trained to predict T_{train} positions. Table 3 reports the average errors at time T_{train} and $2T_{train}$ to measure the ability of predictors to generalise beyond the training regime.

We see that the *Net* models (*ProbNet*, *DispNet*, *PosNet*) perform well using ground-truth positions or the unsupervised tracker outputs (e.g. *PosNet* error for BowlS/HeightS is 2.9/6.4 supervised vs 4.9/6.9 unsupervised), whereas IFS does not handle the transition well (3.3/10.4 to 13.3/23.1) and Linear, Quadratic

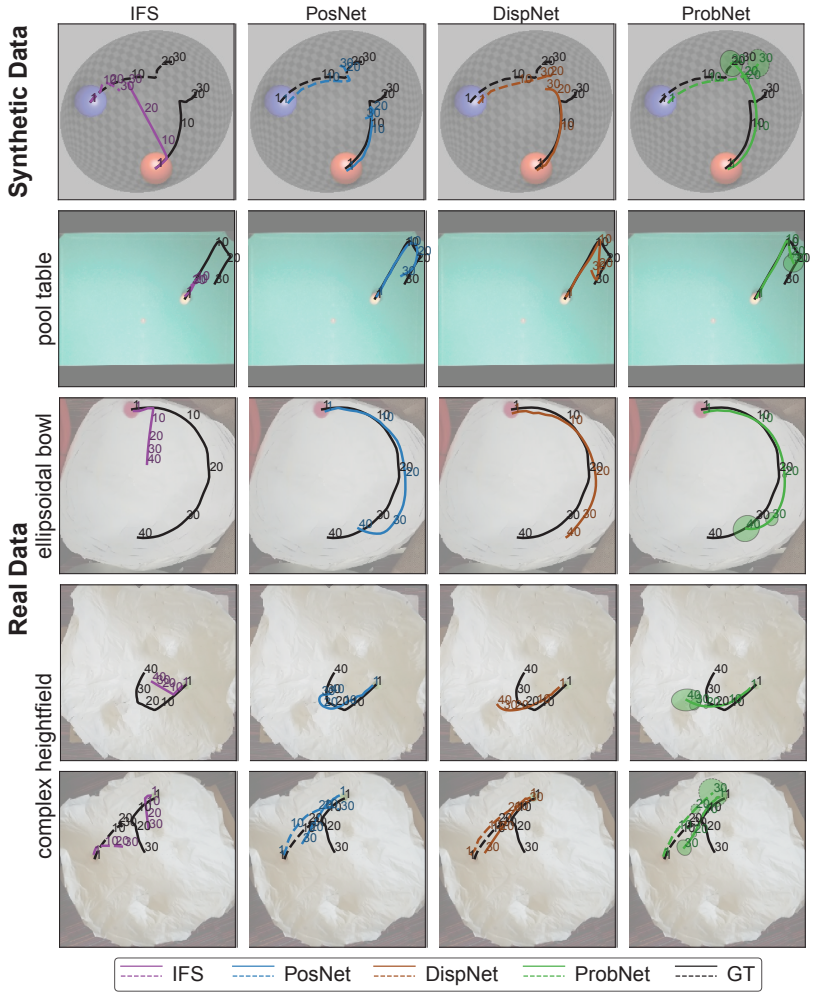


Fig. 5. Qualitative performance comparison for the various methods against ground-truth trajectories Top-to-bottom: two balls colliding on an ellipsoidal bowl; single ball colliding against the walls of a pool table; single ball rolling on an ellipsoidal bow; single ball rolling on complex height-field; and two balls rolling on complex height-field. The top row is on synthetic data, while the other rows are on real-data. The green ellipsoids in the last column show the variance of the predictions estimated by *ProbNet* at selected locations.

and VIN are not competitive. The latest result shows a clear advantage of tensor-based state representations compared to vector based one. This suggests that modelling objects positions is done better by a representation which is spatially distributed. IFS also seems very sensitive to defects in the supplied annotations,

Table 2. Long term predictions compared on synthetic datasets with model trained with ground-truth from simulator All the models (except VIN, Linear, and Quadratic) are given $T_0 = 4$ frames as input and train to predict first T_{train} positions. We report the average pixel error and perplexity for *PosNet* model at two different times. Perplexity, shown in bracket, is defined as $2^{-\mathbb{E}[\log_2(p(x))]}$ where p is the estimated posterior distribution. *State* shows either the carried forward state is a physical quantity (Exp.), or an implicit vector or tensor (Imp.)

			BowlS- T_{train}= 20		HeightS- T_{train}= 20	
			T =T_{train}	2×T_{train}	T_{train}	2×T_{train}
Method	Input	State	With positions from simulator			
Linear	2D pos.	Exp.	61.9	20.1	21.3	61.9
Quadratic	2D pos.	Exp.	11.7	93.1	26.7	126.0
IFS	2D pos.	Exp.	3.3	**8.9**	10.4	27.6
VIN	Visual	Imp.	24.0	30.2	42.6	42.7
PosNet	Visual	Imp.	**1.6**	24.4	7.2	24.6
DispNet	Visual	Imp.	2.5	20.6	7.7	25.8
ProbNet	Visual	Imp.	2.9 (32.1)	21.8 (54.0)	**6.4** (9.5)	**22.5** (12.7)
Method	Input	State	With positions from *unsupervised* tracker			
IFS	2D pos.	Exp.	13.3	**23.6**	23.1	38.3
VIN	Visual	Imp.	24.7	30.3	45.8	48.0
PosNet	Visual	Imp.	4.3	29.9	**6.6**	25.6
DispNet	Visual	Imp.	**3.9**	25.6	6.8	**22.7**
ProbNet	Visual	Imp.	4.9 (6.3)	27.0 (20.6)	6.9 (8.3)	23.3 (13.4)

Table 3. Long term predictions using one ball and real data The table has the same format as Table 2. All models are trained using the unsupervised tracker, input and state are the same as Table 2, and we report pixel error (perplexity) at T.

	PoolR-T_{train} = 15		HeightR- T_{train} = 20		BowlR- T_{train} = 20	
Method	T =T_{train}	2×T_{train}	T_{train}	2×T_{train}	T_{train}	2×T_{train}
IFS	26.0	37.5	48.0	58.1	26.2	39.1
VIN	50.9	40.8	40.2	47.3	33.9	33.0
PosNet	4.6	21.4	**5.6**	29.0	**5.6**	23.0
DispNet	**3.8**	23.6	**5.6**	**28.5**	6.5	**22.6**
ProbNet	4.7(6.3)	**16.3**(11.3)	5.7(5.8)	30.0(22.5)	6.8(6.8)	23.5(13.8)

since its knowledge of the environment is very limited, error correction is very challenging for it.

The main weakness of the *Net* models is that their performance degrades as prediction extends beyond the training horizon $2T_{train}$, whereas IFS generalizes more. At least *ProbNet* explicitly indicates that the model is uncertain when this occurs.

Synthetic vs Real (One Ball Datasets). On real datasets (Table 3), the *Net* models uniformly outperform others at both T_{train} and $2T_{train}$, with errors comparable to the synthetic case. Note that the real datasets in Roll4Real are particularly challenging due to the non-idealities of the surface (e.g. the BowlR surface is slightly elastic and wobbles as the ball rolls).

Table 4. Long term predictions using two balls on real and synthetic data Table layout and measures are the same as Table 2. Models are trained with positions from tracker, input and state are the same as Table 2, and we report pixel error (perplexity) at T.

Method	BowlS 2b.-$T_{train} = 15$				HeightR 2b.-$T_{train} = 15$			
	$T = T_{\text{train}}$		$2{\times}T_{\text{train}}$		T_{train}		$2{\times}T_{\text{train}}$	
IFS	18.4		30.0		15.6		26.6	
VIN	41.3		45.8		45.9		39.8	
PosNet	**5.0**		**13.4**		**5.4**		**12.5**	
DispNet	5.5		24.7		6.2		15.4	
ProbNet	5.6	(7.3)	20.6	(13.7)	6.8	(7.9)	16.9	(12.4)

One vs Multiple Balls (Real and Synthetic Datasets). Finally, we move to cases where the balls are interacting with the environment and with each others due to collisions. This is particularly challenging when no ground-truth is used as multiple object tracking is much harder to achieve in an unsupervised setting than tracking a single object.

As shown in Table 4, the *Net* models still perform well. Due to memory limitations, models were trained for a slightly shorter time span T_{train}; since the corresponding predictions are shorter term, their errors are a little lower than before. Overall, the results show that neither perfect ground-truth annotations nor a very large dataset is required to train a reliable physical extrapolator. Still, we noticed that collisions were difficult to predict in the HeightR dataset (see the bottom row of Fig. 5), probably because such events are rare during training. In contrast, this seems to be much better handled by the models in the synthetic dataset (First row of Fig. 5).

5.3 Unsupervised Physics Interpolation

As in [11], we also study the interpolation problem considering their *InterpNet* configuration. We compare the latter to the extrapolation network *DispNet* trained over a longer horizon $T_{train} = \{30, 40\}$. *InterpNet* has the same architecture has *DispNet* with the difference that, in addition to the first T_0 frames of the sequence, *InterpNet* additionally takes as input the last video frame as well. The first extracted state is used to regress the first T_0 positions as well as the positions at time T_{train}, so that this state is explicitly encouraged to encode information about the last position of the object as well. In Table 5 and Table 6 we see that *InterpNet* managed to reduce the error in most cases. However in this case, compared to results in [11] *InterpNet* performs poorer on synthetic dataset and estimation of the intermediate states seems to be more challenging. Our interpretation is that the imperfect nature of the training data creates several possible path that this model in unable to solve. Finally we also noticed that the heightfield datasets seem to be very challenging as training for longer horizons didn't reduce the error as much as it does on the 'bowl.'

Table 5. Extrapolation vs interpolation: one ball datasets One ball datasets synthetic and real. Models are trained with positions from tracker. Pixel error at different time T.

	PoolR $T_{\text{train}} = 30$			BowlS $T_{\text{train}} = 40$				HeightS $T_{\text{train}} = 40$				BowlR $T_{\text{train}} = 40$				HeightR $T_{\text{train}} = 40$			
T	10	20	30	10	20	30	40	10	20	30	40	10	20	30	40	10	20	30	40
DispNet	3.1	5.6	10.1	3.8	4.0	4.2	4.2	5.2	8.2	13.2	19.1	4.1	5.0	5.5	6.9	4.3	6.6	9.4	12.7
InterpNet	4.5	5.6	3.1	3.8	4.2	4.0	3.8	4.5	6.4	6.5	4.2	6.3	6.5	4.8	3.7	4.0	5.0	4.8	4.3

Table 6. Extrapolation vs interpolation: two balls datasets Two balls datasets synthetic and real. Models are trained with positions from tracker. Pixel error at different time T.

	BowlS 2b $T_{\text{train}} = 30$			HeightR 2b $T_{\text{train}} = 30$		
T	10	20	30	10	20	30
DispNet	4.3	6.9	9.7	5.2	8.9	13.4
InterpNet	4.2	5.0	4.1	6.5	6.9	7.6

6 Conclusions

We presented a method that can learn to track physical objects such as balls rolling on complex terrains using only raw video sequences and no supervision. Combined with recent neural networks that can learn an implicit representation of physics, such a system is able to extrapolate object trajectories over time while accounting for object-environment and object-object interactions. To the best of our knowledge, this is the first time that learning long-term physics extrapolation without access to supervision or handcrafted simulators has been demonstrated. Through an extensive benchmark we also demonstrated the superiority of tensor-based state representation that were able to produce satisfactory results on real data without the need of large datasets.

We also contributed a new dataset, Roll4Real, of real-life video sequences for complex scenarios such as ball rollings on pool tables, bowls, and height-field, showing that all such methods are applicable to the real world. This data will be made publicly available.

In this work we used different colored objects to make them distinguishable, which in practice is one of the main limitation of our work. We plan to address this issue by using same colored objects and build a tracker that would be trained to detect all objects at once removing the need for iterative training.

Finally, we also plan to train the tracker and the extrapolator end-to-end, further improving tracking of multiple objects. We also aim at improving the generalisation of the predictors beyond the training regime; we believe that the key is to factor knowledge about the environment and the object dynamics to allow the models to remember the first better over longer time spans.

Acknowledgements. The authors would like to gratefully acknowledge the support of ERC 638009-IDIU and ERC SmartGeometry StG-2013-335373 grants.

References

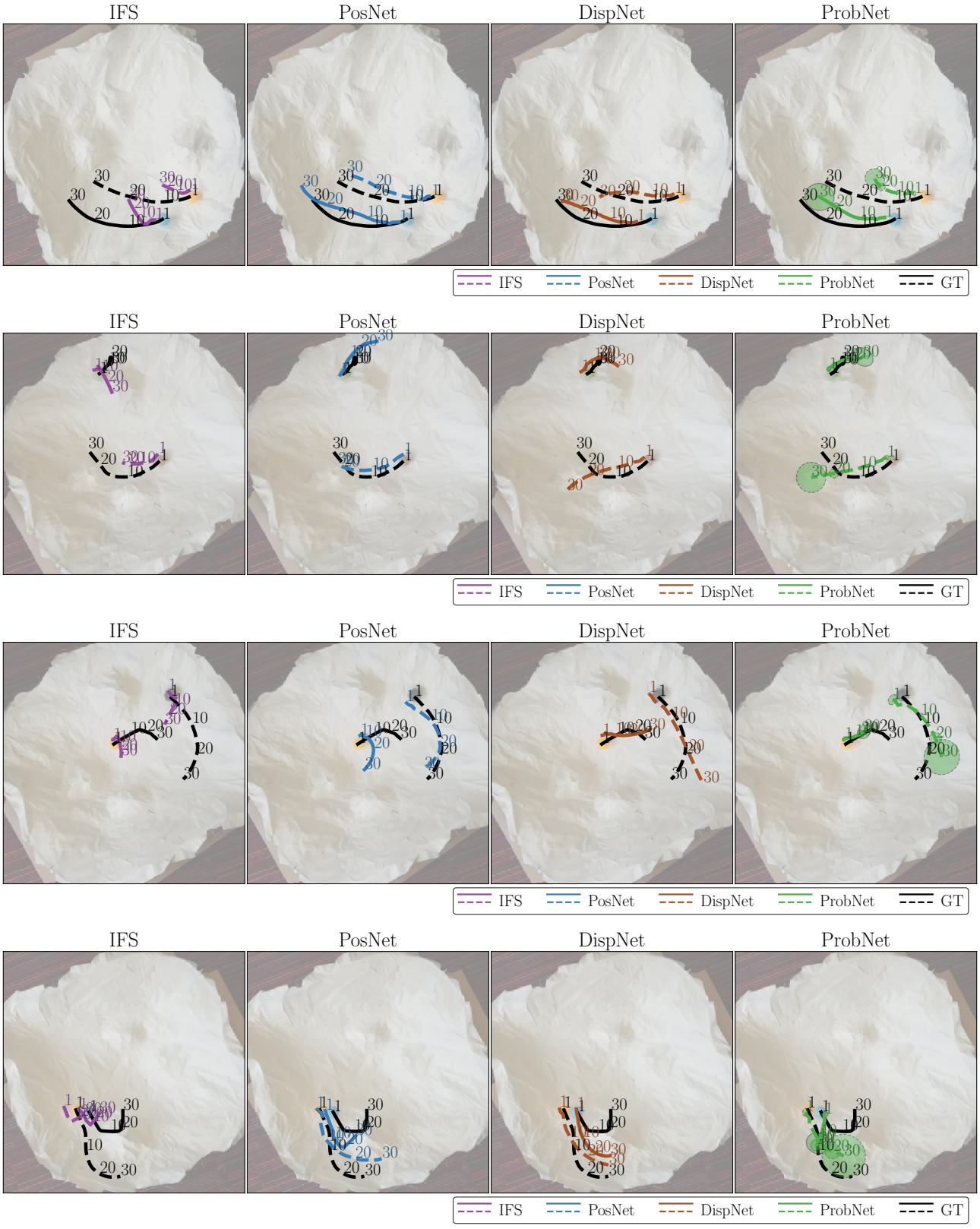
1. Abadi, et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), software available from tensorflow.org
2. Agrawal, P., et al.: Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In: Proc. NIPS. pp. 5074–5082 (2016)
3. Battaglia, P., et al.: Interaction networks for learning about objects, relations and physics. In: Proc. NIPS. pp. 4502–4510 (2016)
4. Battaglia, P., Hamrick, J., Tenenbaum, J.: Simulation as an engine of physical scene understanding. PNAS **110**(45), 18327–18332 (2013)
5. Bhattacharyya, A., et al.: Long-term image boundary prediction. In: Thirty-Second AAAI Conference on Artificial Intelligence. AAAI (2018)
6. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
7. Chang, M.B., et al.: A compositional object-based approach to learning physical dynamics. In: Proc. ICLR (2017)
8. Chiappa, S., et al.: Recurrent environment simulators (2017)
9. Denil, M., et al.: Learning to perform physics experiments via deep reinforcement learning. Deep Reinforcement Learning Workshop, NIPS (2016)
10. Ehrhardt, S., others.: Learning A Physical Long-term Predictor. arXiv e-prints arXiv:1703.00247 (Mar 2017)
11. Ehrhardt, S., et al.: Learning to Represent Mechanics via Long-term Extrapolation and Interpolation. arXiv preprint arXiv:1706.02179 (Jun 2017)
12. Eslami, S.A., et al.: Attend, infer, repeat: Fast scene understanding with generative models. In: Advances in Neural Information Processing Systems. pp. 3225–3233 (2016)
13. Finn, C., et al.: Deep spatial autoencoders for visuomotor learning. In: Robotics and Automation (ICRA), 2016 IEEE International Conference on. pp. 512–519. IEEE (2016)
14. Fragkiadaki, K., et al.: Learning visual predictive models of physics for playing billiards. In: Proc. NIPS (2016)
15. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256 (2010)
16. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks
17. Kansky, K., et al.: Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In: International Conference on Machine Learning. pp. 1809–1818 (2017)
18. Ladický, et al.: Data-driven fluid simulations using regression forests. ACM Trans. on Graphics (TOG) **34**(6), 199 (2015)
19. Lee, A.X., et al.: Stochastic adversarial video prediction. arXiv preprint arXiv:1804.01523 (2018)
20. Lerer, A., Gross, S., Fergus, R.: Learning physical intuition of block towers by example. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48. pp. 430–438 (2016)
21. Li, W., Leonardis, A., Fritz, M.: Visual stability prediction and its application to manipulation. AAAI (2017)
22. Luc, P., Neverova, N., Couprie, C., Verbeek, J., LeCun, Y.: Predicting deeper into the future of semantic segmentation. ICCV (2017)

23. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and learn: unsupervised learning using temporal order verification. In: European Conference on Computer Vision. pp. 527–544. Springer (2016)
24. Monszpart, A., Thuerey, N., Mitra, N.: SMASH: Physics-guided Reconstruction of Collisions from Videos. ACM Trans. on Graphics (TOG) (2016)
25. Mottaghi, R., et al.: Newtonian scene understanding: Unfolding the dynamics of objects in static images. In: IEEE CVPR (2016)
26. Mrowca, D., et al.: Flexible Neural Representation for Physics Prediction. ArXiv e-prints (2018)
27. Novotny, D., et al.: Self-supervised learning of geometrically stable features through probabilistic introspection (2018)
28. Oh, J., et al.: Action-conditional video prediction using deep networks in atari games. In: Advances in Neural Information Processing Systems. pp. 2863–2871 (2015)
29. Ondruska, P., Posner, I.: Deep tracking: Seeing beyond seeing using recurrent neural networks. In: Proc. AAAI (2016)
30. Riochet, R., et al.: IntPhys: A Framework and Benchmark for Visual Intuitive Physics Reasoning. ArXiv e-prints (2018)
31. Sanborn, A.N., Mansinghka, V.K., Griffiths, T.L.: Reconciling intuitive physics and newtonian mechanics for colliding objects. Psychological review **120**(2), 411 (2013)
32. Sanchez-Gonzalez, A., et al.: Graph networks as learnable physics engines for inference and control (2018)
33. Stewart, R., Ermon, S.: Label-free supervision of neural networks with physics and domain knowledge. In: AAAI. pp. 2576–2582 (2017)
34. Thewlis, J., Bilen, H., Vedaldi, A.: Unsupervised learning of object frames by dense equivariant image labelling. In: Advances in Neural Information Processing Systems (NIPS). pp. 844–855 (2017)
35. Tompson, J., et al.: Accelerating Eulerian Fluid Simulation With Convolutional Networks. ArXiv e-print arXiv:1607.03597 (2016)
36. Watters, N., et al.: Visual interaction networks: Learning a physics simulator from video. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 4542–4550. Curran Associates, Inc. (2017)
37. Wu, J., et al.: Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In: Proc. NIPS. pp. 127–135 (2015)
38. Wu, J., et al.: Physics 101: Learning physical object properties from unlabeled videos. In: Proc. BMVC (2016)
39. Wu, J., et al.: Learning to see physics via visual de-animation. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems (NIPS) 30, pp. 153–164. Curran Associates, Inc. (2017)
40. Wu, J., et al.: Learning to see physics via visual de-animation. In: Proc. NIPS (2017)

Supplementary Material for "RELATE: Physically plausible Multi-Object Scene Synthesis Using Structured Latent Spaces"

Extrapolation results (qualitative) on the previously unseen *test* sets from the ROLL4REAL (R4R) dataset. For the corresponding quantitatitive results, please see the tables in the paper.

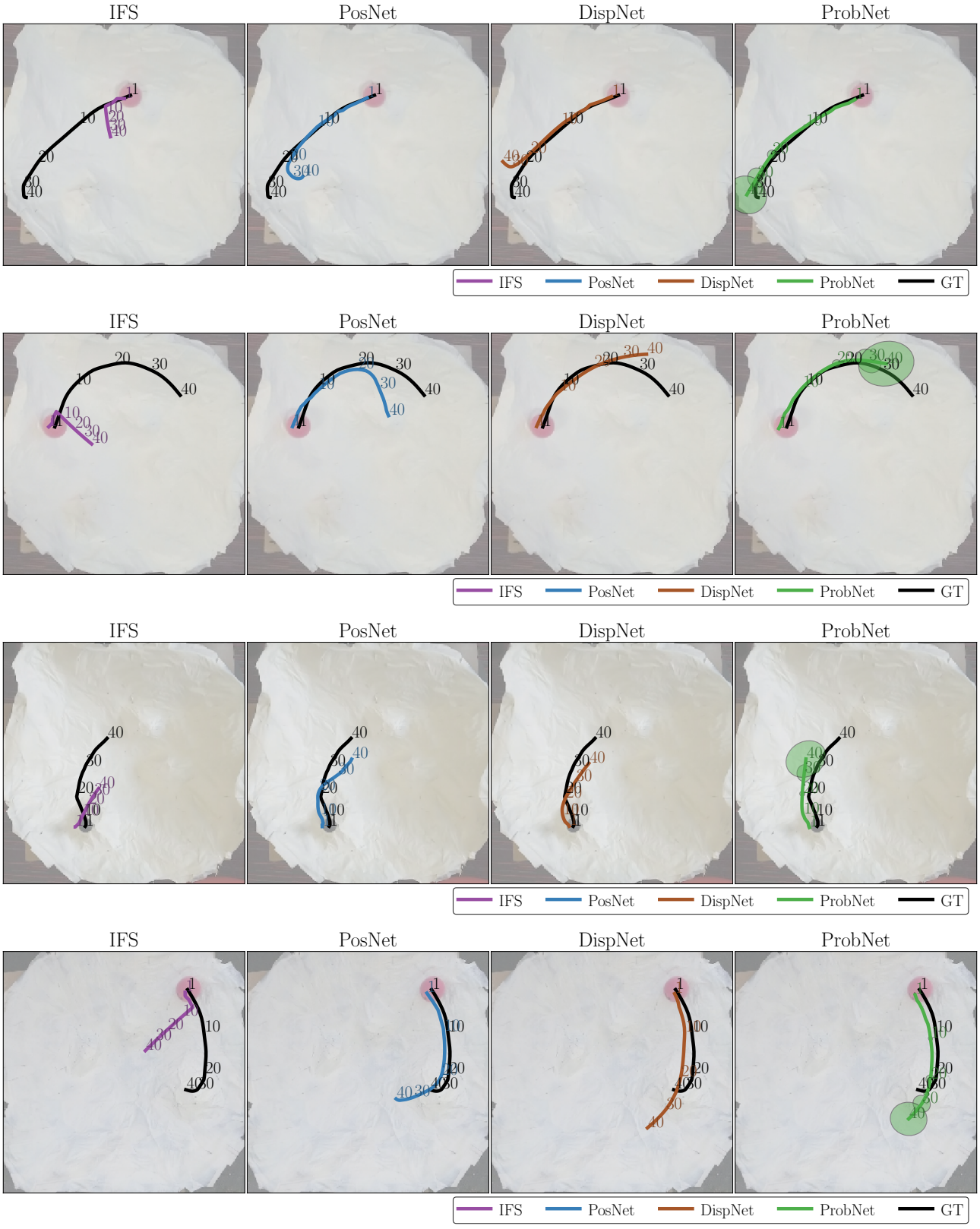
1 Real heightfield 2balls (HEIGHTR 2b.)


IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT


IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT

2 Real heightfield 1ball (HeightR 1b.)


IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT

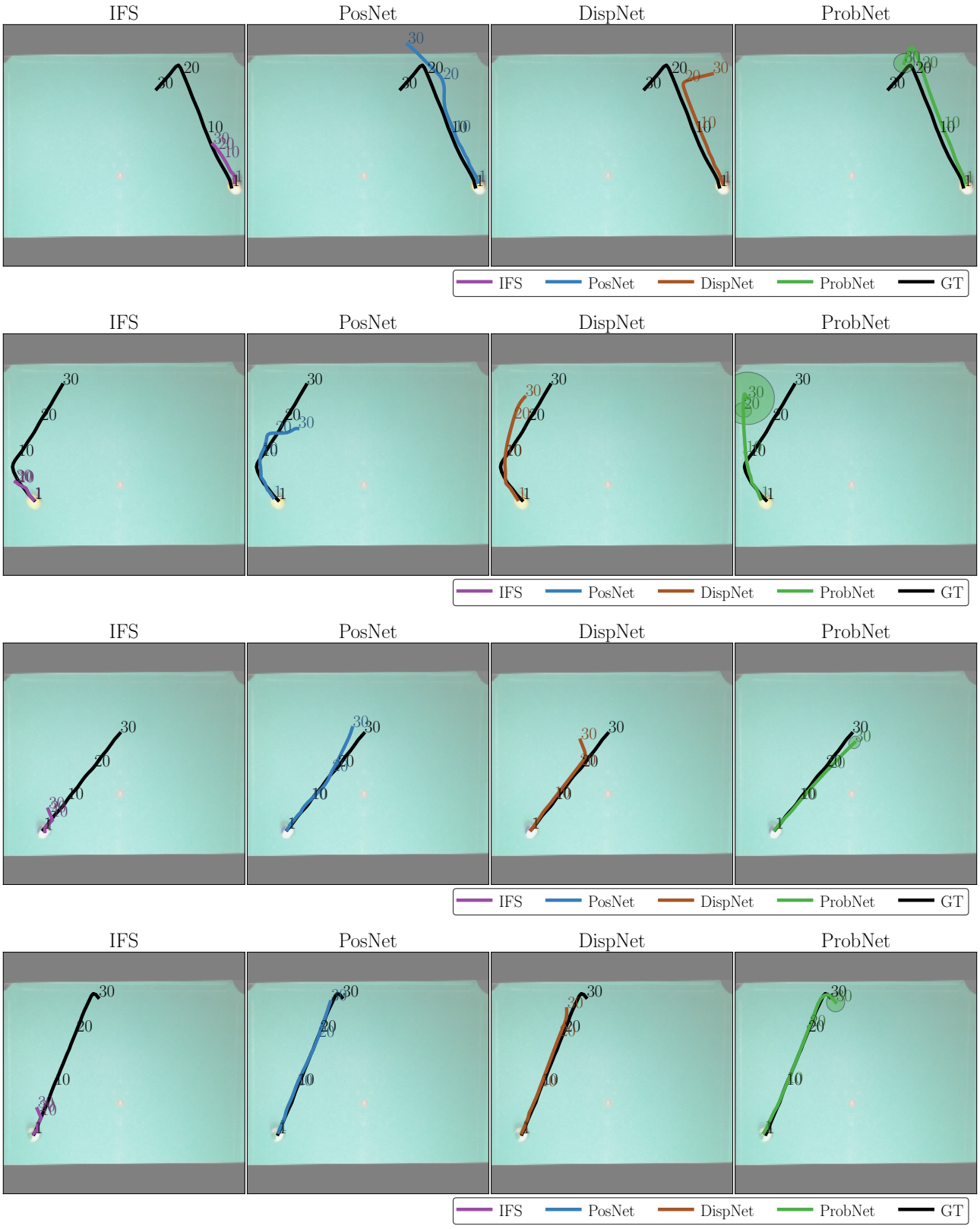

IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT

3 Real ellipsoid 1ball (BowlR 1b.)

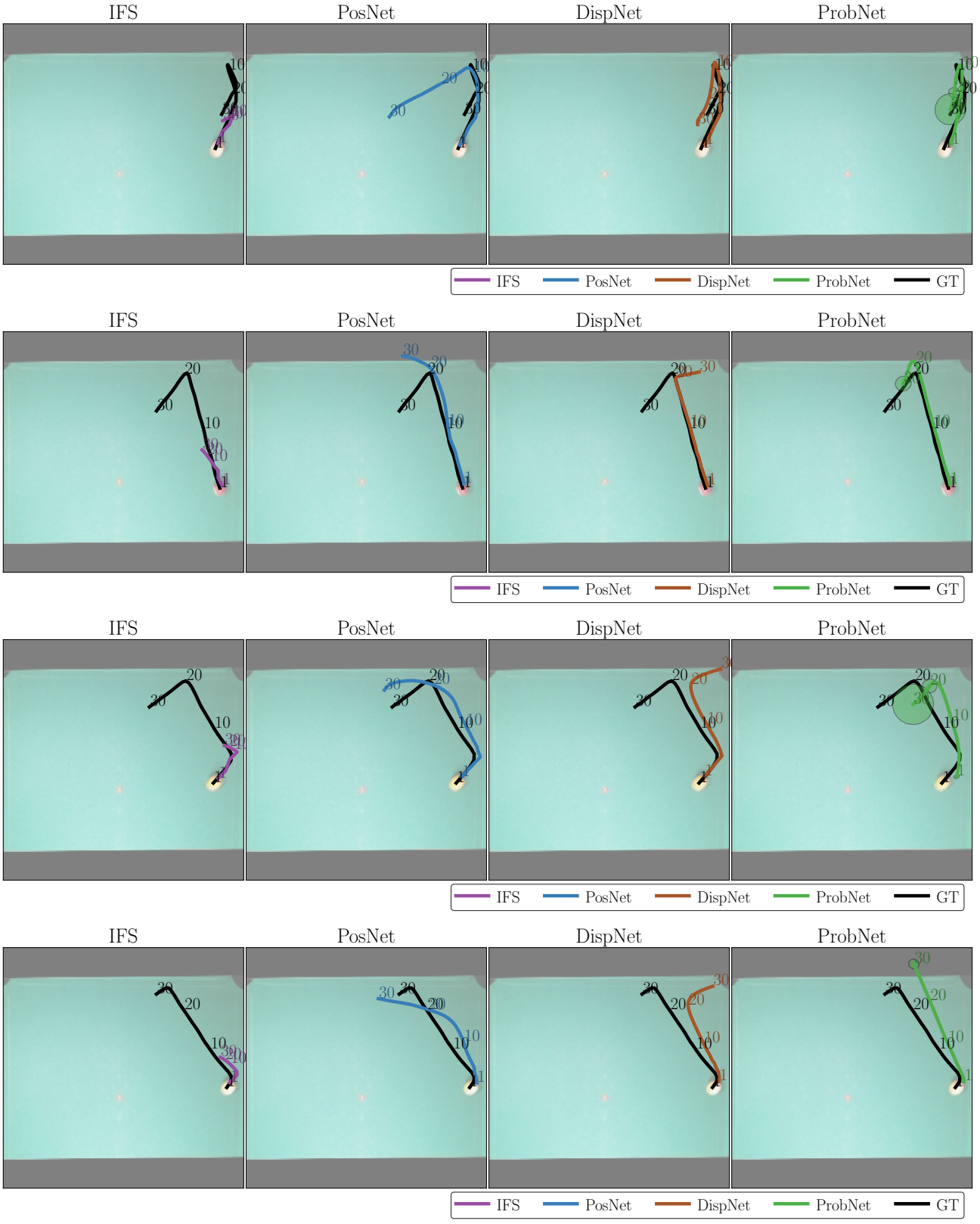

IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT


IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT

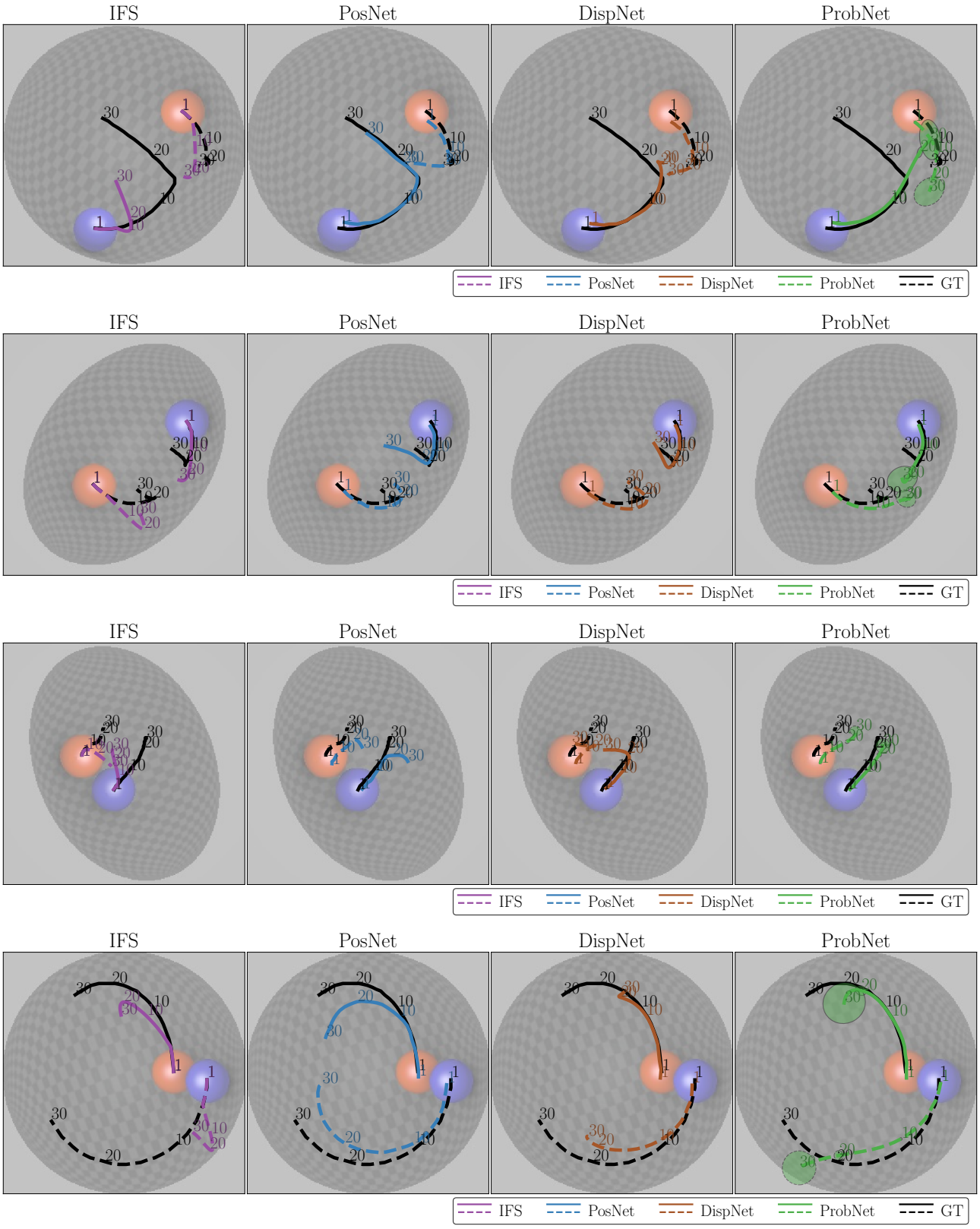
4 Real pool 1ball (PoolR 1b.)

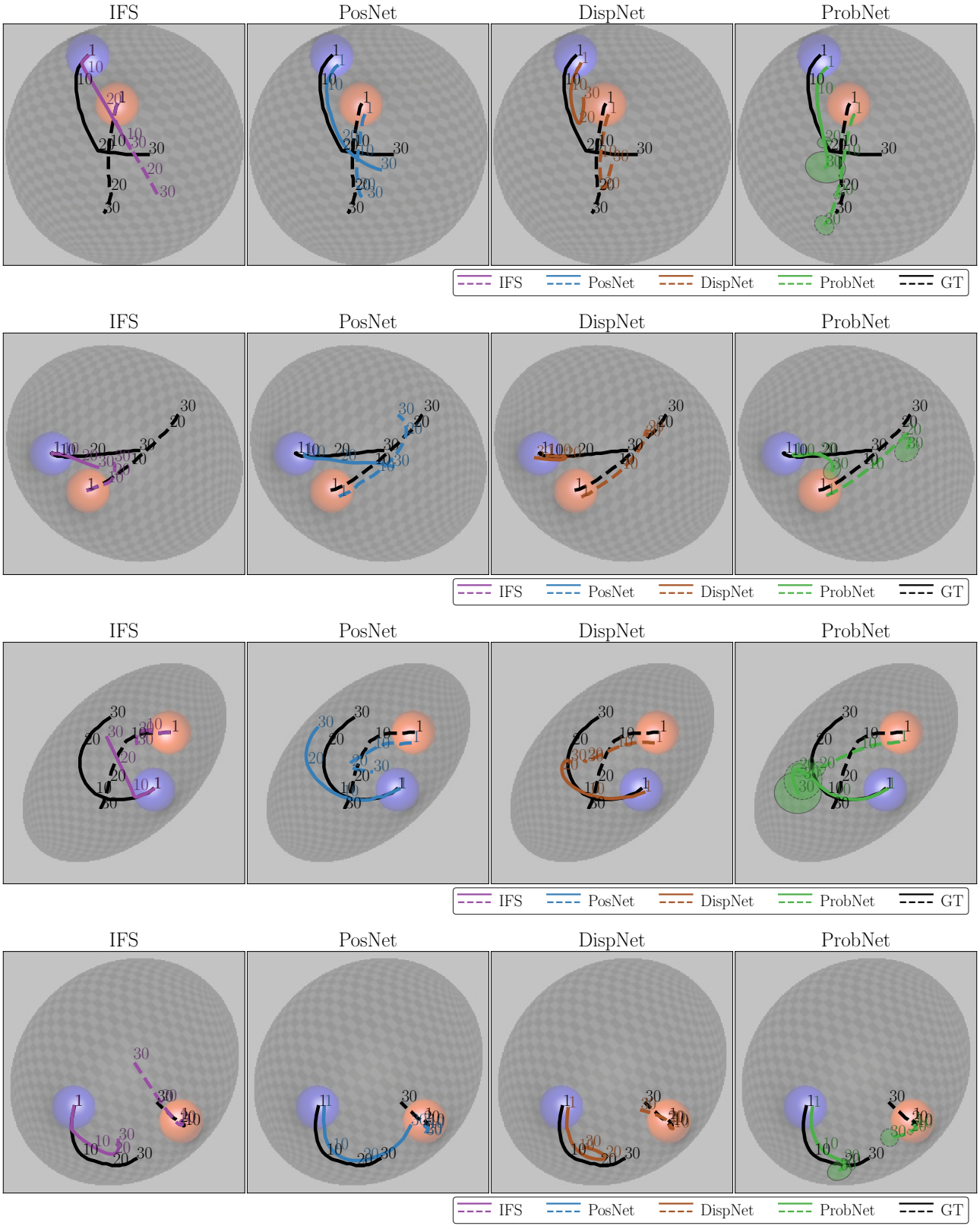


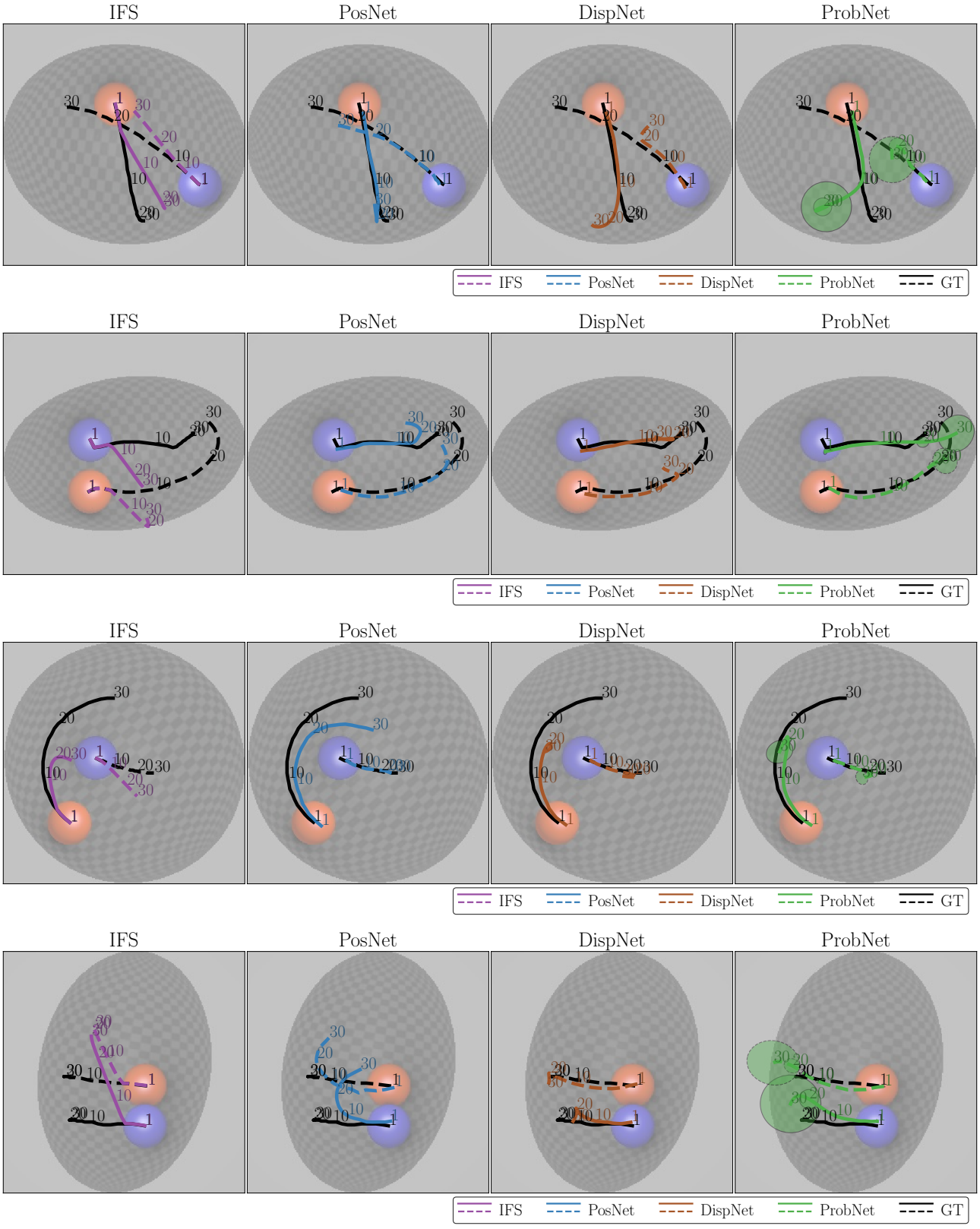

IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
GT


IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
GT

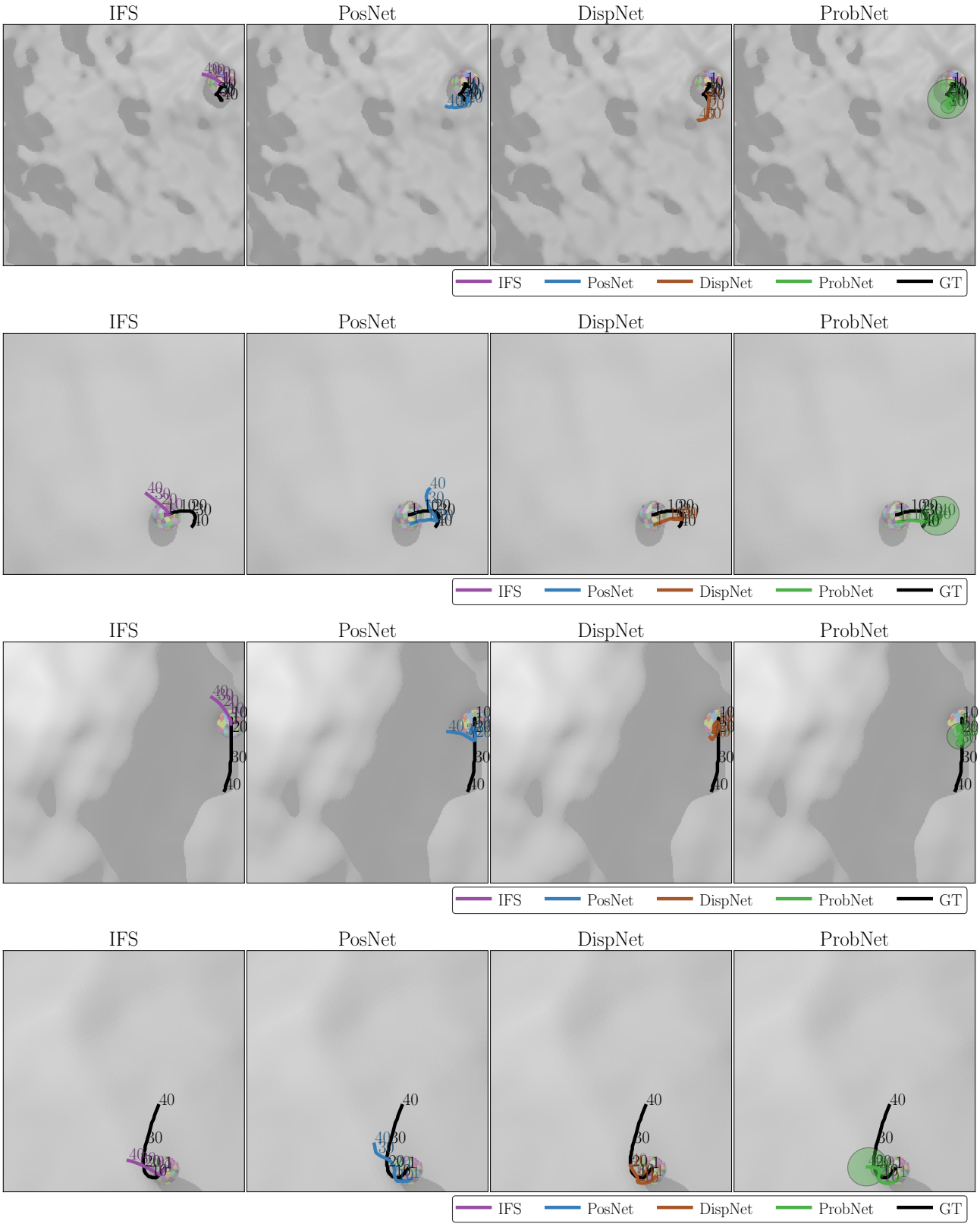
5 Synthetic ellipsoid 2balls (BOWLS 2b.)

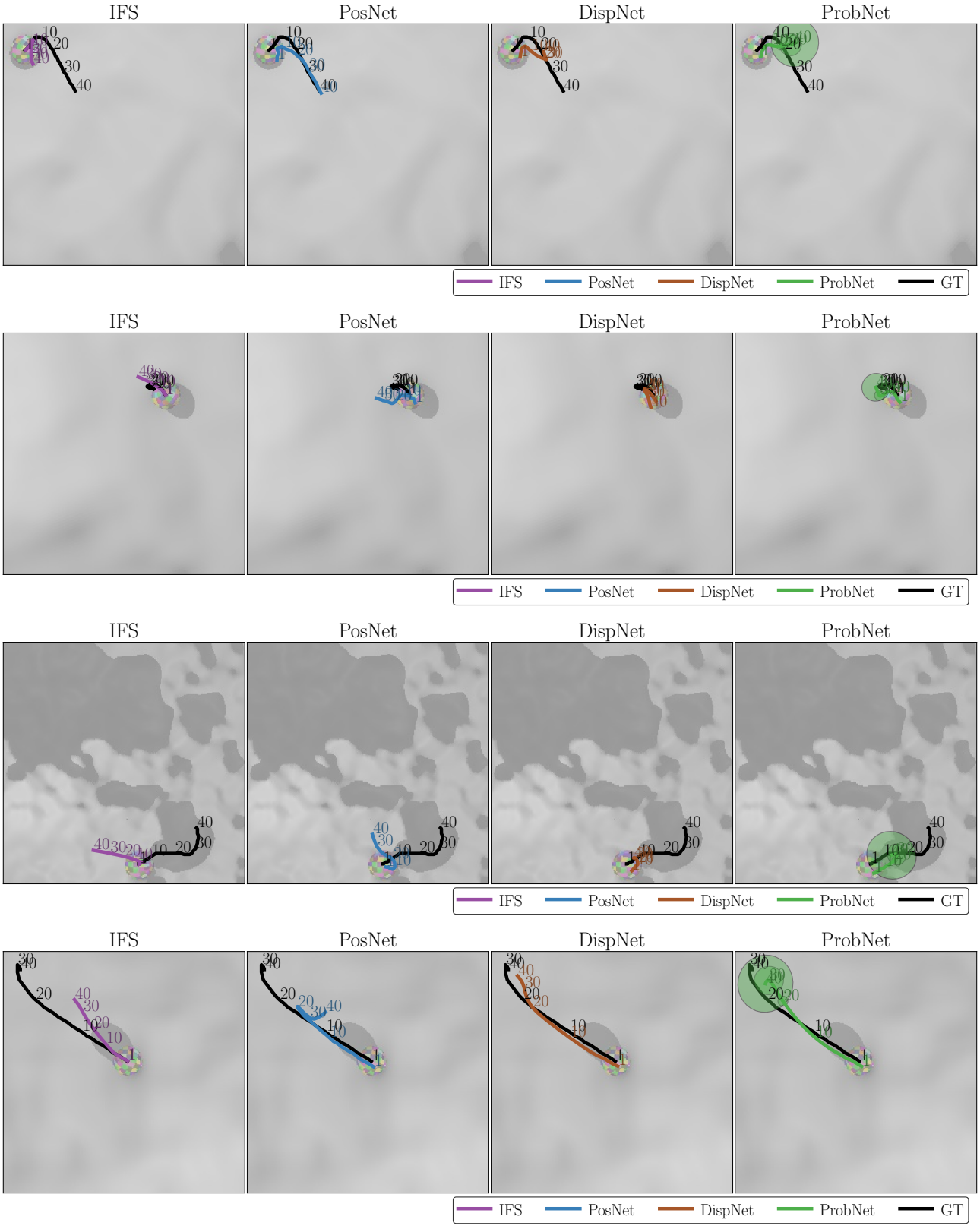



IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
GT


IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
GT

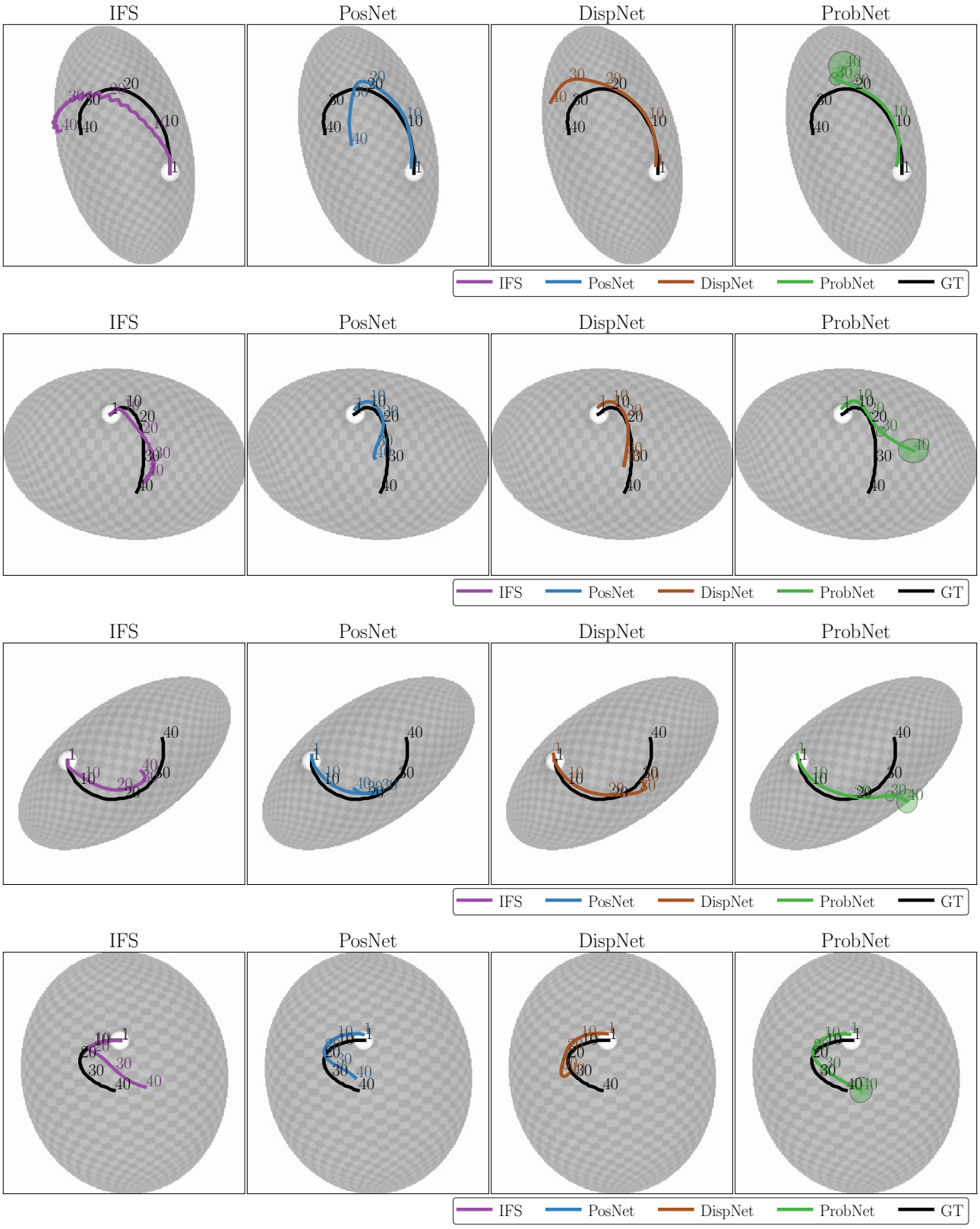
6 Synthetic heightfield 1ball (HeightS 1b.)

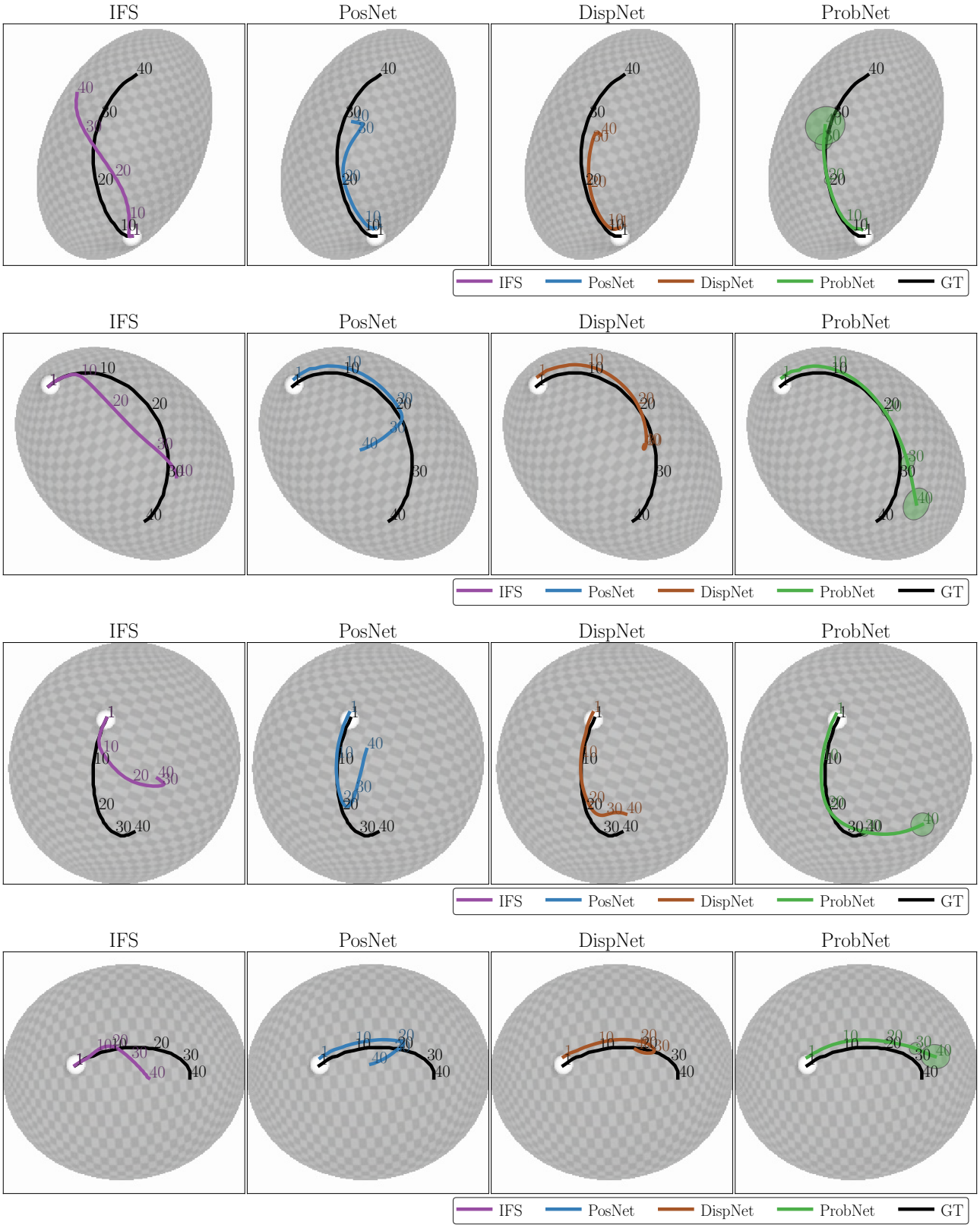


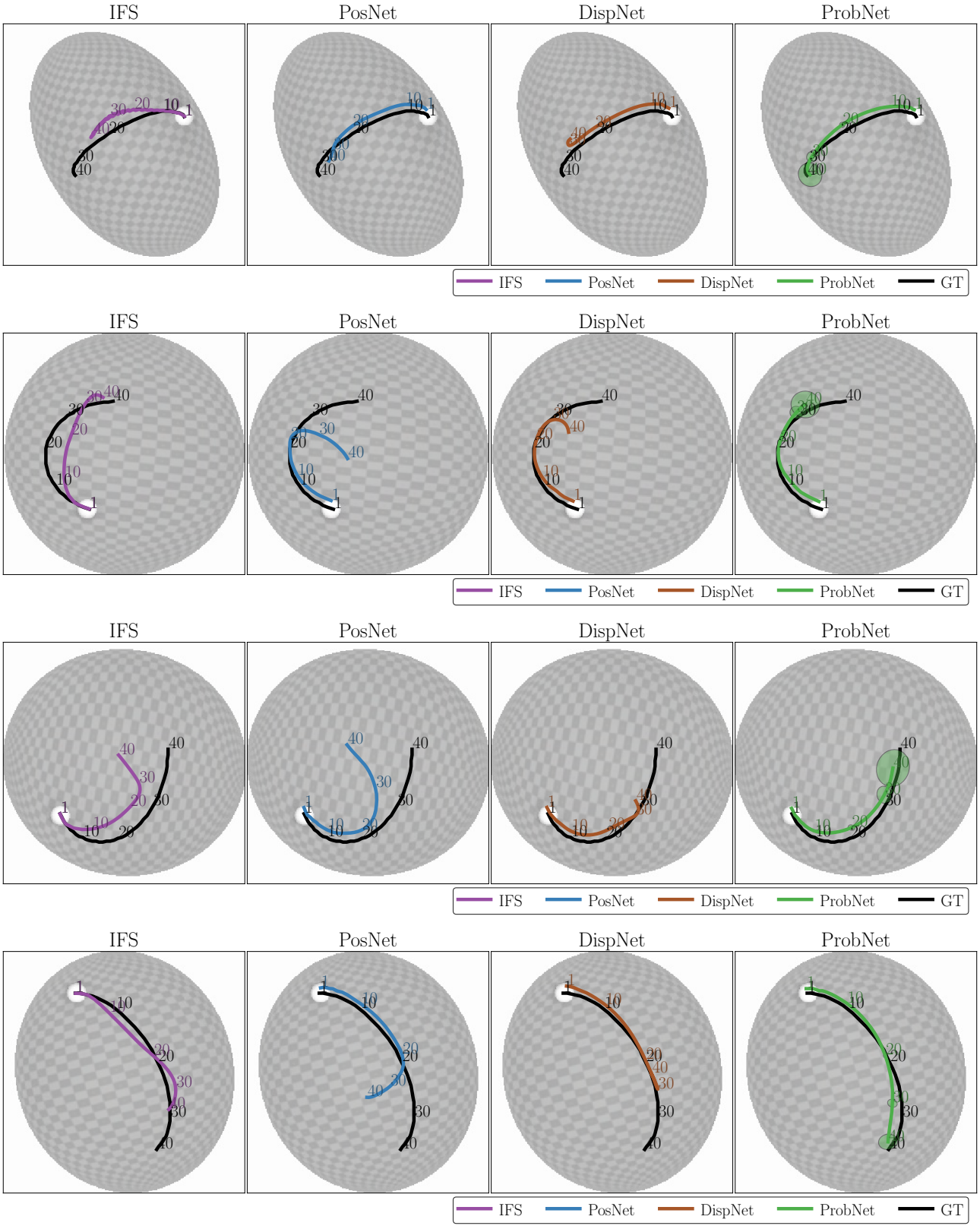

IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
GT


IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
ProbNet
GT

7 Synthetic ellipsoid 1ball (BowlS 1b.)




IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT
IFS
PosNet
DispNet
ProbNet
IFS PosNet DispNet ProbNet GT


IFS
PosNet
DispNet
ProbNet
IFS
PosNet
DispNet
GT

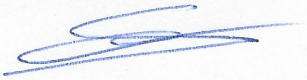
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Unsupervised intuitive physics from visual observations
Publication Status	□Published xAccepted for Publication □Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style
Publication Details	Sebastien Ehrhardt*, Aron Monszpart, Niloy Mitra and Andrea Vedaldi Asian Conference on Computer Vision (2018) *denotes equal contibutions

Student Confirmation

Student Name:	Sebastien Ehrhardt		
Contribution to the Paper	• Contributed to the development of the ideas. • Implementation, evaluation and ablation of the method. • Evaluation of the previous work on the long term prediction task. • Writing paper. • Answer to reviews (rebuttal).		
Signature		Date	27.07.2020.

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof. Andrea Vedaldi		
Supervisor comments		
Signature 	Date	31/07/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 5

Unsupervised intuitive physics from past experiences

In this chapter we try to overcome some of the limitations of the two previous chapters: (i) the representation of physics was internal and abstract and (ii) it needed extensive labelling on data. We also aim to keep long term predictions at the heart of our method following the main thread of this thesis.

We introduce a novel meta-learning architecture to deduce physical properties of objects from past experiences *without* external supervision. Our method starts from an efficient way to summarise past experiences via dynamic images (Bilen et al., 2017) in order to avoid the use of heavy recurrent neural networks. Our architecture uses a meta-learner that abstracts environment properties such as objectness and texture from past experiences. These are then given as input to a propagator that uses this information to extrapolate the motion of one or several objects in different scenarios. We show that the architecture we propose obeys a few desirable properties: it can make long term predictions, scale to bigger images and scale to more objects. We also show how the extracted environment tensor can be used to directly regress object properties such as rigidity.

UNSUPERVISED INTUITIVE PHYSICS FROM PAST EXPERIENCES

Sébastien Ehrhardt[1] **Aron Monzpart**[3] **Niloy Mitra**[2] **Andrea Vedaldi**[1]
[1]Department of Engineering Science, University of Oxford
[2]Department of Computer Science, University College London
[3]Niantic
{hyenal,vedaldi}@robots.ox.ac.uk

ABSTRACT

We consider the problem of learning models of intuitive physics from raw, unlabelled visual input. Differently from prior work, in addition to learning general physical principles, we are also interested in learning "on the fly" physical properties specific to new environments, based on a small number of environment-specific experiences. We do all this in an unsupervised manner, using a meta-learning formulation where the goal is to predict videos containing demonstrations of physical phenomena, such as objects moving and colliding with a complex background. We introduce the idea of summarizing past experiences in a very compact manner, in our case using dynamic images, and show that this can be used to solve the problem well and efficiently. Empirically, we show, via extensive experiments and ablation studies, that our model learns to perform physical predictions that generalize well in time and space, as well as to a variable number of interacting physical objects.

1 INTRODUCTION

Many animals possess an intuitive understanding of the physical world. They use this understanding to accurately and rapidly predict events from sparse sensory inputs. In addition to general physical principles, many animals also learn specific models of new environments as they experience them over time. For example, they can explore an environment to determine which parts of it can be navigated safely and remember this knowledge for later reuse.

Authors have looked at equipping artificial intelligences (AIs) with analogous capabilities, but focusing mostly on performing predictions from instantaneous observations of an environment, such as a few frames in a video. However, such predictions can be successful only if observations are combined with sufficient prior knowledge about the environment. For example, consider predicting the motion of a bouncing ball. Unless key parameters such as the ball's elasticity are known a priori, it is impossible to predict the ball's trajectory accurately. However, after observing at least one bounce, it is possible to infer some of the parameters and eventually perform much better predictions.

In this paper, we are interested in learning intuitive physics in an entirely unsupervised manner, by passively watching videos. We consider situations in which objects interact with scenarios that can only be *partially* inferred from their appearance, but that also contain objects whose parameters cannot be confidently predicted from appearance alone (fig. 1). Then, we consider learning a system that can observe a few physical experiments to infer such parameters, and use this knowledge to perform better predictions in the future.

Our model has three goals. First, it must learn without the use of any external or ad-hoc supervision. We achieve this by training our model from raw videos, using a video prediction error as a loss.

Second, our model must be able to extract information about a new scenario by observing a few experiments, which we formulate as meta-learning. We also propose a simple representation of the experiments based on the concept of "dynamic image" that allows to process long experiments more efficiently than using a conventional recurrent network.

Third, our model must learn a good representation of physics without access to any explicit or external supervision. Instead, we propose three tests to support this hypothesis. (i) We show that the model

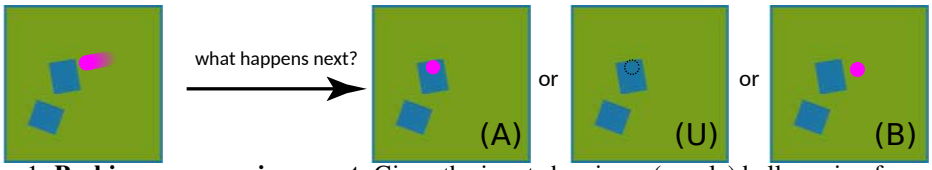


Figure 1: **Probing a new environment.** Given the input showing a (purple) ball moving from right to left, three possible future outcomes are possible: the ball passes above (A), passes under (U), or bounces (B) off the object. This cannot be predicted from appearance alone. Instead, our model learns the property of the object by probing each new environment via a small set of experiments.

can predict far in the future, which is a proxy to *temporal invariance*. (ii) We further show that the model can extend to scenarios that are geometrically much larger than the ones used for training, which is a proxy to *spatial invariance*. (iii) Finally, we show that the model can generalize to several moving objects, which is a proxy to *locality*. Locality and time-space invariance are of course three key properties of physical laws and thus we should expect any good intuitive model of physics to possess them.

In order to support these claims, we conduct extensive experiments in simulated scenarios, including testing the ability of the model to cope with non-trivial visual variations of the inputs. While the data is simpler than a real-world application, we nevertheless make substantial progress compared to previous work, as discussed in section 2. We do so by learning from passive, raw video data a good model of dynamics and collisions that generalizes well spatially, temporally, and to a variable number of objects. The scalability of our approach, via the use of the dynamic image, is also unique. Finally, we investigate the problem of learning the parameters of new scenarios on the fly via experiences and we propose an effective solution to do so.

2 Related work

A natural way to represent physics is to manually encode every object parameter and physical property (e.g., mass, velocity, positions) and use supervision to make predictions. This has been widely used to represent and propagate physics (Wu et al., 2015; 2016; Battaglia et al., 2016; Chang et al., 2017; Mrowca et al., 2018; Sanchez-Gonzalez et al., 2018). If models like Wu et al. (2015; 2016) also estimate environment parameters, these works rely on a physics engine that assumes strong priors about the scenario, while our approach does not require such constraint.

Inspired by the recent successes of Convolutional Neural Networks (CNNs (Krizhevsky et al., 2012)) and their application to implicit representation of dynamics (Ondruska & Posner, 2016; Oh et al., 2015; Chiappa et al., 2017; Bhattacharyya et al., 2018), researchers (Watters et al., 2017; Ehrhardt et al., 2019; Kansky et al., 2017) have tried to base their approaches on visual inputs. They learn from several frames of a scene to regress the next physical state of a system. In general these approaches learn an implicit representation of physics (Ehrhardt et al., 2019; Watters et al., 2017) as a tensor state from recurrent deep networks.

Such models are mostly supervised using ground-truth information about key physical parameters (e.g., positions, velocities, density) during training. While these approaches require an expensive annotation of data, other works have tried to learn from unsupervised data as well. Researchers have successfully learned unsupervised models either through active manipulation (Agrawal et al., 2016; Denil et al., 2016; Finn et al., 2016a), using the laws of physics (Stewart & Ermon, 2017), using dynamic clues and invariances (Greff et al., 2017; van Steenkiste et al., 2018) or features extracted from unsupervised methods (Finn et al., 2016b; Ehrhardt et al., 2018). Fragkiadaki et al. (2016) also used an unsupervised system like ours, however, they assumed the rendering system and the number of objects to be known, which we do not, which prevented the internal representation to fail over time. Perhaps most related to our approach is the work of Wang et al. (2018), where the model is learnt using future image prediction on a simple synthetic task and then transferred to real world scenarios. They also demonstrate long-term dynamic predictions, however they did not generalize to different background/number of balls.

In other works, models are taught to answer simple qualitative questions about a physical setup, such as: the stability of stacks of objects (Battaglia et al., 2013; Lerer et al., 2016; Li et al., 2017; Groth et al., 2018), the likelihood of a scenario (Riochet et al., 2018), the forces acting behind a scene (Wu

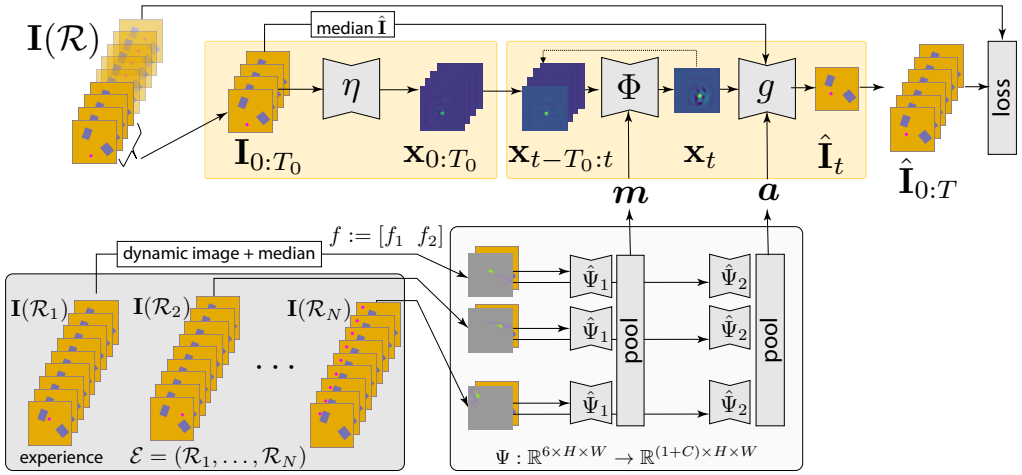


Figure 2: **Overview of our method.** Ψ (bottom right block) acts as a meta-learning module. It takes as input past experiments compressed into dynamic images alongside with the median image and learn to optimise $\boldsymbol{m}$ and $\boldsymbol{a}$ to minimise the final loss. η (top-left yellow block) extracts states $\mathbf{x}_{0:T_0}$ which are carried forward with the auto-regressive predictor Φ. Finally g renders frame $\hat{\mathbf{I}}_t$ from $\mathbf{x}_t$.

et al., 2017; Mottaghi et al., 2016) or properties of objects through manipulation (Agrawal et al., 2016; Denil et al., 2016). Other papers compromise between qualitative and quantitative predictions and focus on *plausibility* (Tompson et al., 2016; Ladický et al., 2015; Monszpart et al., 2016).

Finally similar problems can be found in online learning settings (Nagabandi et al., 2019b;a). These works also use meta-learning but, differently from ours, allow their models to learn and adapt at test time from feedbacks.

3 Method

We describe our model, illustrated in fig. 2, starting from the input and output data.

A *scenario* S is a physical environment that supports moving objects interacting with it. In this paper, we take as scenarios 2.1D environments containing obstacles and we consider rolling balls as moving objects. Hence, interactions are in the form of bounces. Formally, a scenario is defined over a lattice $\Omega = \{0, \ldots, H-1\} \times \{0, \ldots, W-1\}$ and is specified by a list of obstacles $\mathcal{S} = \{(O_j, b_j),\ j = 1, \ldots, K\}$. Here $O_j \subset \Omega$ is the shape of an obstacle and $b_j \in \{B, A, U\}$ is a flag that tells whether the ball bounces against it (B), passes above it (A), or passes under it (U). Obstacles do not overlap.

A *run* is a tuple $\mathcal{R} = (\mathcal{S}, \mathbf{y})$ associating a scenario $\mathcal{S}$ with a trajectory $\mathbf{y} = (y_t \in \Omega, t = 0, \ldots, T-1)$ for the ball (this is trivially extended to multiple balls). Scenarios and runs are sensed visually. The symbol $\mathbf{I}(\mathcal{S}) : \Omega \to \mathbb{R}^{3\times H\times W}$ denotes the image generated by observing a scenario with no ball and $\mathbf{I}_t(\mathcal{R}) = \mathbf{I}(\mathcal{S}, y_t)$ is the image generated by observing the ball at time t in a run. The symbol $\mathbf{I}(\mathcal{R}) = (\mathbf{I}_t(\mathcal{R}), t = 0, \ldots, T-1)$ denotes the collection of all frames in a run, which can be thought of as a video.

We are interested in learning intuitive physics with no explicit supervision on the object trajectories, nor an explicit encoding of the laws of mechanics that govern the motion of the objects and their interactions and collisions with the environment. Hence, we cast this problem as predicting the video $\mathbf{I}(\mathcal{R})$ of the trajectory of the objects given only a few initial frames $\mathbf{I}_{0:T_0}(\mathcal{R}) = (\mathbf{I}_0(\mathcal{R}), \ldots, \mathbf{I}_{T_0-1}(\mathcal{R}))$, where $T_0 \ll T$.

We are not the first to consider a similar learning problem, although most prior works do require some form of external supervision, which we do not use. Here, however, we consider an additional key challenge that the images $\mathbf{I}_{0:T_0}(\mathcal{R})$ *do not* contain sufficient information to successfully predict the long-term objects' motion. This is because these few frames tell us nothing about the *nature* of the obstacles in the scenario. In fact, under the assumptions that obstacles of type B, A and U have similar or even the identical appearance, it is not possible to predict whether the ball will bounce, move above, or move under any such obstacle. This situation is representative of an agent that needs

to operate in a new complex environment and must learn more about it, from past experiences, before it can do so reliably.

Thus, we consider a modification of the setup described above in which the model can experience each new scenario for a while, by observing the motion of the ball, before making its own predictions. Formally, an *experience* is a collection of N runs $\mathcal{E} = (\mathcal{R}_1, \dots, \mathcal{R}_N)$ all relative to the same scenario $\mathcal{S}$ with each a different trajectories $\mathbf{y}_1, \dots, \mathbf{y}_N$. By observing such examples, the model must determine the nature of the obstacles and then use this information to correctly predict the motion of the ball in the future. We cast this as the problem of learning a mapping

$$\Omega \ : \ (\mathbf{I}_{0:T_0}(\mathcal{R}),\ \mathbf{I}(\mathcal{E})) \ \longmapsto \ \mathbf{I}(\mathcal{R}), \tag{1}$$

where $\mathbf{I}(\mathcal{E}) = (\mathbf{I}(\mathcal{R}_1), \dots, \mathbf{I}(\mathcal{R}_N))$ are the videos corresponding to the runs in the experience. We will call $\mathcal{R}$ the *prediction run* to distinguish it from the experience runs $\mathcal{E}$.

3.1 A META-LEARNING PERSPECTIVE

The setup we have described above can be naturally described as meta-learning. Namely, eq. (1) can be thought of as incorporating a "local" learning rule $\mathcal{M}$ that maps the experience $\mathcal{E}$ to a scenario-specific predictor $\hat{\Omega}$ on the fly: $\hat{\Omega}(\cdot) = \Omega(\cdot, \mathbf{I}(\mathcal{E})) = \mathcal{M}[\mathbf{I}(\mathcal{E})]$. Hence $\mathcal{M}$ must extract from the experience as much information as possible about the underlying scenario and transfer it to the scenario-specific predictor $\hat{\Omega}$. In order to learn $\mathcal{M}$, we consider *meta-samples* of the type $(\mathcal{S}, \mathcal{R}, \mathcal{E})$ comprising a scenario $\mathcal{S}$, a prediction run $\mathcal{R}$ and N experience runs $\mathcal{E}$. Given a dataset $\mathcal{D}$ of such meta-samples, meta-learning searches for the mapping $\mathcal{M}$ that minimizes the error on the prediction runs:

$$\mathcal{M}^* = \operatorname*{argmin}_{\mathcal{M}} \frac{1}{|\mathcal{D}|} \sum_{(\mathcal{S},\mathcal{R},\mathcal{E})\in\mathcal{D}} \ell(\mathbf{I}(\mathcal{R}),\ \hat{\Omega}(\mathbf{I}_{0:T_0}(\mathcal{R}))), \quad \hat{\Omega} = \mathcal{M}[\mathbf{I}(\mathcal{E})]. \tag{2}$$

3.2 COMPRESSED DYNAMIC EXPERIENCES

Concretely, we parameterise the scenario-specific predictor $\hat{\Omega}(\cdot; \boldsymbol{w}, \boldsymbol{m})$ using parameters $\boldsymbol{w}$, the weights a deep neural network, which is fixed and scenario-independent, and $\boldsymbol{m}$, which is scenario-specific. The latter is extracted by a network $\boldsymbol{m} = \Psi_1(\mathbf{I}(\mathcal{E}))$ from the experience videos $\mathbf{I}(\mathcal{E})$. Since we expect $\boldsymbol{m}$ to provide information about the nature of the obstacles in the scenario, we let $\boldsymbol{m} \in \mathbb{R}^{1\times H\times W}$ be a tensor with the same spatial resolution as the scenario and interpret it as an "obstacle mask".

Given that the function Ψ_1 takes as input a number of video sequences, one may think to implement it as a recurrent neural network; however, recurrent networks scale badly, especially in our meta-learning context — to the point that we had difficulties in even running such a model at a non-trivial scale. Instead, we propose to construct Ψ_1 based on a compact representation of the experience which leads to a much more efficient design.

For this, we use the concept of *dynamic image* (Bilen et al., 2016), which encodes a video as a weighted average of its frames: $f_1(\mathbf{I}_0, \dots, \mathbf{I}_{T-1}) = \sum_{t=0}^{T-1} \alpha_t \mathbf{I}_t$ with $\alpha_t = \sum_{i=t}^{T-1} \frac{2(i+1)-T-1}{i+1}$. Since the dynamic image is only sensitive to changes in the video, we complement it by computing also the *median image* $f_2(\mathbf{I}_0, \dots, \mathbf{I}_{T-1}) = \operatorname{median}_{t=0,\dots,T-1} \mathbf{I}_t$ and combine the two functions f_1 and f_2 in a single representation f by stacking the respective outputs along the channel dimension. With this design, we can rewrite the map Ψ_1 as follows:

$$\boldsymbol{m} = \Psi_1(\mathbf{I}(\mathcal{E})) = \operatorname*{maxpool}_{n=1,\dots,N} \hat{\Psi}_1(f(\mathbf{I}(\mathcal{R}_n))). \tag{3}$$

Here the map $\hat{\Psi}_1 : \mathbb{R}^{6\times H\times W} \to \mathbb{R}^{1\times H\times W}$, which can be implemented as a standard CNN, takes as input the dynamic/median image and produces as output the obstacle mask $\boldsymbol{m}$. The pooling operator summarizes the information extracted from multiple runs into a single mask $\boldsymbol{m}$.

We also consider a second similar map $\hat{\Psi}_2 : \mathbb{R}^{6\times H\times W} \to \mathbb{R}^{C\times H\times W}$ to extract an "appearance" tensor $\boldsymbol{a}$. The latter helps the generator to render obstacles above or underneath the moving objects as needed. The response of this function is also max-pooled over runs, but channel-wise: $[\operatorname{pool}_{k=1,\dots,N} \boldsymbol{a}_k]_{cvu} = [\boldsymbol{a}_{k(c)}]_{cvu}$ where $k(c) = \operatorname{argmax}_{k=1,\dots,N} \sum_{vu} [\boldsymbol{a}_k]^2_{cvu}$.

In practice, $\hat{\Psi}_1$ and $\hat{\Psi}_2$ are implemented as a single neural network $\Psi : \mathbb{R}^{6\times H\times W} \to \mathbb{R}^{(1+C)\times H\times W}$ where $\hat{\Psi}_1$ is the first output channel and $\hat{\Psi}_2$ the others.

Optional obstacle mask supervision. In the experiments, we show that the map $\hat{\Psi}$ can be learned automatically without any external supervision. We contrast this with supervising $\hat{\Psi}_1$ with an oracle rendition of the obstacle map. To this end, we define the tensor $\boldsymbol{m}_{\text{gt}}$ for a scenario $\mathcal{S}$ to be the indicator mask of whether a pixel contains a solid obstacles (including the perimetral walls) and then minimize the auxiliary loss $\ell(\boldsymbol{m}_{\text{gt}}, \Psi_1(\mathbf{I}(\mathcal{E}))) = \|\boldsymbol{m}_{\text{gt}} - \Psi_1(\mathbf{I}(\mathcal{E}))\|^2$.

3.3 Auto-regressive predictor

The predictor Φ is designed as an RNN that takes as input T_0 past *states* $\mathbf{x}_{t-T_0:t} = (\mathbf{x}_{t-T_0}, \dots, \mathbf{x}_{t-1})$ and outputs a new state $\mathbf{x}_t$. Each state variable is in turn a distributed representation of the physical state of the system, in the form of a "heatmap" $\mathbf{x}_t \in \mathbb{R}^{H\times W}$. Considering T_0 past states allows the model to auto-regressively represent the dynamics of the system if so learning chooses to do.

The predictor also takes as input the scenario representation $\boldsymbol{m}$ given by eq. (3). The first T_0 state variables are initialized from observations $\mathbf{I}_{0:T_0}(\mathcal{R})$ via an initialization function $\eta : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^{H\times W}$. We thus have

$$\mathbf{x}_{0:T_0} = [\eta(\mathbf{I}_t(\mathcal{R}))]_{t=0:T_0} \qquad \text{(initialization)} \tag{4}$$

$$\mathbf{x}_t = \Phi(\mathbf{x}_{t-T_0:t}, \boldsymbol{m}) \qquad \text{(auto-regressive prediction)} \tag{5}$$

In short, this model estimates recursively the evolution of the system dynamics from the visual observations of the first T_0 samples.

Conditional generator. Variable $\mathbf{x}_t$ contains information about the state of the moving objects (balls). This is then converted into a prediction of the corresponding video frame, combining also the appearance tensor $\boldsymbol{a}$ and the median of the first T_0 images in the sequence $\hat{\mathbf{I}} = \text{median}_{t=0,\dots,T_0-1}\, \mathbf{I}_t$. This is formulated as a conditional generator network $\hat{\mathbf{I}}_t = g(\mathbf{x}_t, \boldsymbol{a}, \hat{\mathbf{I}})$.

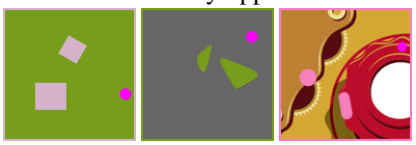
Video reconstruction loss. Next, we discuss the nature of the loss equation 2. Owing also to the static background, the *conditional* video generation task is relatively simple provided that the dynamics are estimated correctly. As a consequence, the generated videos are likely to closely approximate the ground truth ones, so the loss function ℓ in equation 2 does not require to be complex. In our experiments, we combine the L^2 image loss with the *perceptual* loss of (Johnson et al., 2016). The latter is obtained by passing the ground-truth and predicted images through the first few layers of a pre-trained deep neural network e, VGG-16 (Simonyan & Zisserman, 2015) in our case, and then comparing the resulting encodings in L^2 distance. The overall loss is then given by $\ell(\mathbf{I}_t, \hat{\mathbf{I}}_t) = \lambda_{\mathbf{I}}\|\mathbf{I}_t - \hat{\mathbf{I}}_t\|^2 + \lambda_p\|e(\mathbf{I}_t) - e(\hat{\mathbf{I}}_t)\|^2$ (details in the sup. mat.). The perceptual loss is robust to small shifts in the image reconstructions that may arise from imperfect physics predictions and favors reconstructing a sharp image; by comparison, the L^2 loss alone would result in blurrier images by regressing to the mean. In practice, trading-off the L^2 and perceptual losses results in qualitatively better images as well as in better trajectory prediction.

State space integrity. An issue with the recursion equation 5 is that the state $\mathbf{x}_t$ may, in the long run, falls outside the space of meaningful states, leading to a catastrophic failure of the predictor. This is especially true for predictions longer than the ones used for training the model. In order to encourage the recursion to maintain the integrity of the state over time, we add a self-consistency loss $\|\mathbf{x}_t - \eta(\mathbf{I}_t(\mathcal{R}))\|^2$. Here the network η is the same that is used to extract the state space from the initial frames of the video sequence in equation 4 which weights are fixed. In practice, we find that, by using this additional loss, our recursion learns to maintain the integrity of the state nearly indefinitely.

4 Experimental setup

Data. A scenario $\mathcal{S}$ (see embedded figure) is generated by sampling a certain number of obstacles of different types m_i and shapes O_i (either rectangular or custom), placing them at random locations and orientations on a 2D board. We consider boards with either two rectangular obstacles (denoted R2), a random number from 3 to 4 rectangular obstacles (R4) or two curved obstacles (C). Scenarios are rendered by painting the background and a wall around it. Then, all obstacle are painted in a solid colour, randomly picked from a fixed palette to ensure suffi-

cient contrast. The background is also painted in solid color (general case) or by using a texture image (only curved shapes, denoted C+T). Crucially, there is no correlation between an obstacle's type and its shape, location and color, so its type cannot be inferred by appearance alone. Runs $\mathcal{R}$ are generated by placing one or more dynamic objects ("balls") with a random initial location and a random momentum oriented towards the obstacles, simulating the 2.1D physics, and rendering the balls as circles with a fixed color (see fig. A1 in supp. mat. for more examples).



For each scenario $\mathcal{S}$, we sample $N + 1$ runs, using the first as a prediction run and the others as experience runs, forming triplets $(\mathcal{S}, \mathcal{R}, \mathcal{E})$. Unless otherwise stated, we set $N = 7$ and let each run evolve for 60 frames, including for the experience runs. Thanks to our compact representation, we only use 200MB for 7 experiences for backpropagation during training. Using a recurrent network requires an amount of memory larger in proportion with the length of an experience, i.e. 60-fold, so 12GB for 64×64 boards, which wouldn't allow the reccurrent architecture to fit into memory. Using shorter runs is also not possible, as they tend to contain no collisions.

For training, samples $(\mathcal{S}, \mathcal{R}, \mathcal{E})$ are drawn in an on-line fashion and are thus unlimited. For testing, we use a fixed set of 200 samples. We generate boards of size 64×64 and 128×128, but use the latter only for testing purposes to evaluate generalization. Unless otherwise stated, the board size is 64×64.

Evaluation metrics. We report the video prediction error as average L_2 image loss. In order to assess the ability of the method to predict good trajectories, we note that blobs tend to emerge in the heatmaps $\mathbf{x}_t$ in correspondence of the tracked objects. Hence, we handcrafted a simple blob detector $h(\mathbf{x}_t)$ that detects the blobs contained in the heatmaps $\mathbf{x}_t$ (see sup. mat.). We then report the number of detected blobs vs the actual number of moving objects and, for each, the distance in pixel space of the predicted blob center and ground-truth object center, averaged over the different scenarios. For each experiment we report mean and standard deviation across all sampled scenarios.

Baselines. We compare with three baselines. The first is a version of the Interaction Networks (Battaglia et al., 2016), trained with perfect knowledge of the object locations, background and object generation. The baseline works directly on the object positions and regresses object positions. The ground truth obstacle map is given as input and transformed into a vector thanks to a pre-trained VGG-16 architecture. We also reimplemented the RLVN network of Wang et al. (2018), which has a vector representation bottleneck, and compared video prediction error results. Note that this baseline wouldn't adapt to bigger board. Our last baseline amounts to running the ground-truth physics simulator after removing the obstacles from the board (and thus results in perfect predictions for trajectories that do not hit solid obstacles).

Implementation details. Networks Φ and η share very similar auto-encoder type architecture, the network Ψ uses a U-Net-like architecture (Ronneberger & Fischer, 2015). As stated in Wang et al. (2018) intuitively Ψ would better preserve appearance while the structure of Φ would loose some appearance structure and put an inductive bias on dynamic predictions. g is a fully-convolutional 6-layers stack. Our implementation uses TensorFlow 1.9 (Abadi et al., 2015) and the Adam optimizer (Kingma & Ba, 2014) with learning rate 10^{-4} and Xavier initialization (Glorot & Bengio, 2010). We used a batch size of 10 samples $(\mathcal{S}, \mathcal{R}, \mathcal{E})$. Models are first trained for 110,000 iterations using only the L^2 and self-consistency losses for eq. (2), and then optionally fine-tuning for further 1,000 iterations using the perceptual loss as well. All models are trained on R2 with board size 64^2 and tested *without any fine-tuning* on different board types and sizes (see Tab. 1,2,3,A.3). For the C+T scenarios, we first train the model using flat colour R2 scenarios and then fine-tune for 55,000 iterations using the textured data. Unless otherwise specified, models are trained in a fully unsupervised fashion. Full details can be found in the sup. mat.

5 RESULTS

Full system. Table 1 rows 1-4 report the prediction accuracy of our system on the 200 simulated test sequences using one moving object. In the table, we distinguish three durations: $T_0 = 4$ is the number of frames to bootstrap the prediction of a test runs, $T_{\text{train}} = 20$ is the length of prediction runs observed during meta-training and $T_{\text{test}} = kT_{\text{train}}$ is the length of the runs observed during meta-testing, where $k = 1, 3, 5$. We test both 64×64 boards (which is the same size used for training)

Table 1: **Predicting one moving object.** Obst. is the obstacle type (R2, R4, C). The test board size can be either 64×64 (same as in training) or 128×128 and we consider using either no supervision or obstacle supervision. We test the average prediction error at $T_{\text{test}} = 20, 60, 100$ well above the duration $T_{\text{train}} = 20$ observed during training. Position errors were normalized by the board size diagonal. Video errors were normalized to board size 64^2 where larger. L^2 loss on positions is used for (14)-(15).

					$T_{\text{test}} = T_{\text{train}} = 20$			$T_{\text{test}} = 3 \times T_{\text{train}}$			$T_{\text{test}} = 5 \times T_{\text{train}}$		
No.	Obst.	Sup.	Test brd. size	Train loss	# obj.	Vid. L_2	Pos. err.	# obj.	Vid. L_2	Pos. err.	# obj.	Vid. L_2	Pos. err.
(1)	R2	None	64^2	L^2	1.0±0.2	2.5±2.6	.036±.096	0.3±0.5	4.6±3.3	.404±.241	0.1±0.3	4.7±3.3	.526±.180
(2)	R2	None	128^2	L^2	0.7±0.4	1.7±1.3	.179±.236	0.1±0.3	1.7±1.1	.484±.228	0.0±0.1	1.7±1.1	.531±.209
(3)	R2	None	64^2	Percep.	1.0±0.2	2.5±2.9	.028±.073	1.0±0.2	5.3±4.0	.145±.148	1.0±0.2	5.5±4.0	.286±.168
(4)	R2	None	128^2	Percep.	0.9±0.4	1.7±1.4	.104±.170	1.0±0.4	2.0±1.4	.227±.179	1.0±0.4	2.0±1.4	.320±.172
(5)	R2	Obst.	64^2	Percep.	1.0±0.1	3.0±3.3	.018±.023	1.0±0.2	5.7±4.4	.112±.086	1.0±0.3	5.8±4.2	.231±.141
(6)	R2	Obst.	128^2	Percep.	8.3±3.3	2.4±1.9	.323±.064	26.5±6.4	5.6±4.0	.355±.054	25.8±6.3	5.8±4.1	.363±.133
(7)	C	None	64^2	Percep.	1.0±0.1	2.9±3.5	.040±.086	1.1±0.3	5.4±4.1	.175±.160	1.0±0.3	5.4±4.0	.265±.155
(8)	C	None	128^2	Percep.	0.9±0.4	1.7±1.4	.104±.170	1.0±0.4	2.0±1.4	.227±.179	1.0±0.4	5.6±4.6	.320±.172
(9)	C	Obst	64^2	Percep.	1.0±0.1	2.9±3.1	.024±.072	1.0±0.1	5.7±4.2	.128±.114	1.0±0.2	5.5±4.1	.247±.141
(10)	C+T	None	64^2	Percep.	1.0±0.1	3.4±2.9	.017±.044	1.0±0.2	6.1±3.4	.094±.076	1.0±0.2	6.2±2.9	.222±.147
(11)	C+T	None	128^2	Percep.	1.0±0.1	.675±.685	.012±.033	1.0±0.2	1.4±.814	.057±.065	0.9±0.3	1.4±.750	.121±.088
					Ablation: removing the median and dynamic images, respectively (see text)								
(12)	R2	None	64^2	Percep.	1.0±0.3	4.0±4.0	.074±.115	0.8±0.5	5.1±4.0	.302±.202	0.6±0.5	5.0±3.8	.357±.212
(13)	R2	None	64^2	Percep.	1.4±0.6	3.2±3.4	.054±.062	1.8±0.8	5.5±4.2	.268±.151	1.7±0.8	5.6±4.2	.326±.136
					Comparison: Interaction Network (14-15), RLVN (16) and Ground Truth Simulator minus Obstacles (17-18)								
(14)	R2	Full	128^2	L^2 Pos.	-	-	.038±.027	-	-	.166±.088	-	-	.331±.152
(15)	C	Full	128^2	L^2 Pos.	-	-	.038±.031	-	-	.171±.095	-	-	.349±.203
(16)	R2	-	64^2	-	-	3.6±3.2	-	-	5.3±3.3	-	-	5.5±3.5	-
(17)	R2	-	64^2	-	-	-	.060±.099	-	-	.229±.224	-	-	.224±.209
(18)	C	-	64^2	-	-	-	.055±.099	-	-	.219±.217	-	-	.232±.228

and larger 128×128 boards to test spatial generalization. We also compare training using the L^2 or the L^2+perceptual loss for video prediction supervision. We report the number of detected objects (which should be 1 in this experiment), the video prediction error, and the position prediction error.

We find that switching from the L^2 to the L^2+perceptual loss for training (rows 1 vs 3 and 2 vs 4) reduces significantly the trajectory prediction error and prevents the object from disappearing. This, however, results to slightly larger L_2 video prediction error compared to empty boards predicted in the long run by models trained with L^2 loss only. We also find that generalization through time is excellent: prediction errors accumulate over time (as it is unavoidable), but the recursion evolves without collapsing well beyond T_{train} (qualitatively we found that the recursion can be applied indefinitely). This should be contrasted with prior work on intuitive physics (Fragkiadaki et al., 2016) where recursion was found to be unstable in the long term.

Spatial generalization to larger boards (rows 2,4,6) is also satisfactory if the perceptual loss is used. However, we did notice the emergence of a few artifacts in this case.

In rows 5-6 we use external supervision to train the obstacle map predictor as suggested in section 3.2 (the rest of the system is still unsupervised). Supervision improves the trajectory prediction accuracy for boards of the same size as the ones seen in training. However, generalization to larger boards is poorer, suggesting that explicit supervision causes the model to overfit somewhat.

Table 2: **Importance of experience.** For the unsupervised model, we report the trajectory prediction error at $T = 40$ (rows 1-3) and the obstacle mask L_2 prediction error (rows 4-6) for different obstacle types. The number of runs N in the experiences is varied from 0 to 50. For the obstacle mask prediction error, we also report the trivial baseline $\|\boldsymbol{m}_{\text{gt}} - \boldsymbol{m}_{\text{all-on}}\|_2$ as 'All-ON' (see text). Position errors were normalized by the board size (64^2) diagonal. $T_{\text{train}} = 20$.

No	Err.	Obst.	All-ON	$N = 0$	1	2	5	7	10	20	50
(1)	Pos.	R2	—	.462±.136	.144±.157	.108±.136	.083±.128	.077±.120	.071±.114	.070±.116	.070±.104
(2)	Pos.	R4	—	.502±.151	.189±.169	.158±.164	.122±.155	.110±.149	.114±.130	.110±.154	.114±.152
(3)	Pos.	C	—	.459±.124	.161±.172	.125±.158	.099±.139	.099±.146	.099±.144	.101±.144	.106±.143
(4)	Obs.	R2	13.2±11.1	24.6±16.4	10.3±9.5	7.6±8.4	4.2±5.4	3.8±4.7	3.5±3.3	3.6±3.5	3.8±3.7
(6)	Obs.	R4	16.8±13.3	23.6±15.1	14.1±10.9	11.5±9.9	7.7±8.0	7.4±7.3	7.3±7.2	7.4±7.2	8.4±8.3
(5)	Obs.	C	9.4±8.9	27.8±15.8	10.1±8.2	7.0±7.0	5.4±5.6	5.3±5.2	5.1±5.2	5.7±5.6	6.0±5.9

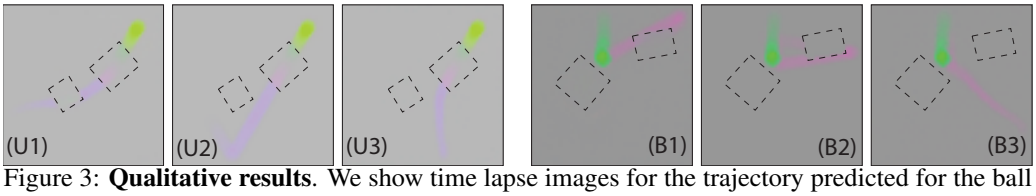


Figure 3: **Qualitative results**. We show time lapse images for the trajectory predicted for the ball colliding with obstacles of type U (permeable; left) and B (solid; right). We test three variants of the model: (1) the model supervised with ground-truth obstacle masks, (2) the unsupervised model and (3) the model after suppressing the dynamic image that summarizes the experience (ablation). Qualitatively, (1,2) produce plausible predictions but (3) does not.

Rows 7-11 show results for more complex shapes (C) and appearances (C+T). Prediction accuracy is only slightly worse than with the simpler obstacles (e.g. rows 3 vs 7 or 9 vs 10).

Ablation. We suppress respectively the median image (row 12) and dynamic image (row 13) in the feature extractor f summarizing the experiences. We notice a sensible deterioration of performance compared to row 3, suggesting that both components are important.

Comparison. Compared to the fully-supervised IN (rows 14-15), our unsupervised method is highly competitive, especially for longer simulation times (rows 4,8). The method also outperforms (except for very long simulations where drift prevails) the ground-truth simulator that ignores the obstacles (rows 16,17). Finally we show that our method surpasses RLVN (Wang et al., 2018)(row 16), qualitatively, this method produces similar results to (13) and the object tends to disappear when encounting an object.

The importance of experience. We investigate how much information the model extracts from the experiences $\mathcal{E}$ by ablating the number of runs N in it. For the special case $N = 0$, since there are no runs, we generate an pseudo experience run by copying the first frame $\mathbf{I}_0(\mathcal{R})$ of the prediction run $\mathcal{R}$.

In table 2, we report the trajectory prediction accuracy for different obstacle types (row 1-3). We also test the ability to predict the obstacle mask $\boldsymbol{m}_{\text{gt}}$ (defined in section 3). Since the model is unsupervised, the learned mask $\boldsymbol{m}$ will not in general be an exact match of $\boldsymbol{m}_{\text{gt}}$ (e.g. the mask range is arbitrary); hence, after fixing all the parameters in the model, we train a simple regression network that maps $\boldsymbol{m}$ to $\boldsymbol{m}_{\text{gt}}$ and report the prediction accuracy of the latter on the test set (see sup. mat. for details). As a point of comparison, we also report the error $\|\boldsymbol{m}_{\text{gt}} - \boldsymbol{m}_{\text{all-on}}\|_2$ of the trivial baseline that predict a mask $\boldsymbol{m}_{\text{all-on}}$ where all objects are highlighted, regardless of their flag.

We note that there is a very large improvement when we move from zero to one experience runs, and a smaller but non-negligible improvement until $N = 10$. Furthermore, comparing the results with predicting $\boldsymbol{m}_{\text{all-on}}$ shows that the system can tell from the experiences which obstacles are solid and which are not.

Table 3: **Predicting the obstacles: supervision vs no supervision.** The table reports the obstacle mask prediction error (L_2) for a network trained with or without supervision for the obstacles (see text). The last column shows a multi-ball predictor operated with 3 balls in each run. $T_{\text{train}} = 20$.

Supervision	R2	R4	C	C+T	R2 (3 balls)	C+T → R2
Obstacle	$2.7_{\pm 4.4}$	$6.3_{\pm 7.8}$	$3.5_{\pm 4.9}$	$2.6_{\pm 3.9}$	$2.6_{\pm 3.9}$	$7.1_{\pm 9.2}$
None	$3.8_{\pm 4.7}$	$7.4_{\pm 7.3}$	$5.3_{\pm 5.2}$	$4.0_{\pm 5.2}$	$3.9_{\pm 3.5}$	$7.6_{\pm 8.9}$

Supervised obstacle regression. Table 3 compares our unsupervised method to using full obstacle map supervision in order to predict the obstacle map $\boldsymbol{m}_{\text{gt}}$ from the experiences $\mathcal{E}$. For the unsupervised system, the obstacle map is estimated as explained in the paragraph above. As expected, supervised learning achieves a lower error, but the unsupervised method is still much better than the trivial baseline of table 2. Finally, we observe that for network fine-tuned on C+T and evaluated back on R2 (C+T→R2 in Table 3) obstacle predictions results remain competitive with results on R2 only while object appearance and backgrounds drastically changed in the new setting.

Multiple moving objects. We also test whether the system trained with a single moving object can generalize to multiple ones. In table A.3 (see sup. mat.) we show that the network can simulate the motion properly until balls collide, after which they merge (see videos in the sup. mat.). This indicates that, just as physical laws, the rules learned by the model are local. We also train a model (scenario

R2) showing it from 1 to 3 balls in each run. This model is able not only to correctly handle ball collisions, but is also able to generalize correctly to several more objects on the board. Furthermore, as shown in table 3, this model still predicts correctly the obstacle masks from experiences, despite the act that the latter are much more crowded.

6 CONCLUSIONS

We have demonstrated a system that can learn an intuitive model of physics in an unsupervised manner. Differently from most prior works, our system is also able to learn on-the-fly and very efficiently some physical parameters of new scenarios from a few experiences. Prediction results are strong, competitive with fully-supervised models, and predictors generalize well over time space, and an arbitrary number of moving objects. Our next challenge is to apply the system to more difficult scenarios, including non-frontal views of the physical worlds. For this, we plan to learn a function to summarize past experiences in a more general manner than the dynamic image could.

Acknowledgements. The authors would like to gratefully acknowledge the support of ERC 638009-IDIU and ERC SmartGeometry StG-2013-335373 grants.

REFERENCES

Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

P. Agrawal et al. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In *Proc. NeurIPS*, pp. 5074–5082, 2016.

P. Battaglia et al. Interaction networks for learning about objects, relations and physics. In *Proc. NeurIPS*, pp. 4502–4510, 2016.

P.W. Battaglia, J. Hamrick, and J.B. Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 110(45):18327–18332, 2013.

A. Bhattacharyya et al. Long-term image boundary prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI, 2018.

H. Bilen et al. Dynamic image networks for action recognition. In *IEEE CVPR*, 2016.

M. B. Chang et al. A compositional object-based approach to learning physical dynamics. In *Proc. ICLR*, 2017.

S. Chiappa et al. Recurrent environment simulators. In *Proc. ICLR*, 2017.

M. Denil et al. Learning to perform physics experiments via deep reinforcement learning. *Deep Reinforcement Learning Workshop, NIPS*, 2016.

S. Ehrhardt et al. Unsupervised intuitive physics from visual observations. *Proc. ACCV*, 2018.

S. Ehrhardt et al. Taking visual motion prediction to new heightfields. *Computer Vision and Image Understanding (CVIU)*, 2019.

C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, 2016a.

C. Finn et al. Deep spatial autoencoders for visuomotor learning. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 512–519. IEEE, 2016b.

K. Fragkiadaki et al. Learning visual predictive models of physics for playing billiards. In *Proc. NeurIPS*, 2016.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

K. Greff et al. Neural expectation maximization. *Proc. NeurIPS*, 2017.

O. Groth et al. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *eccv*, 2018.

J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *Proc. ECCV*, 2016.

K. Kansky et al. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *International Conference on Machine Learning*, pp. 1809–1818, 2017.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *Proc. ICLR*, 2014.

A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NeurIPS*, pp. 1097–1105, 2012.

Ladický et al. Data-driven fluid simulations using regression forests. *ACM Trans. on Graphics (TOG)*, 34(6):199, 2015.

A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pp. 430–438, 2016.

W. Li, A. Leonardis, and M. Fritz. Visual stability prediction and its application to manipulation. *AAAI*, 2017.

A. Monszpart, N. Thuerey, and N. Mitra. SMASH: Physics-guided Reconstruction of Collisions from Videos. *ACM Trans. on Graphics (TOG)*, 2016.

R. Mottaghi et al. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *IEEE CVPR*, 2016.

D. Mrowca et al. Flexible Neural Representation for Physics Prediction. *Proc. NeurIPS*, 2018.

Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *iclr*, 2019a.

Anusha Nagabandi et al. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *iclr*, 2019b.

J. Oh et al. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, pp. 2863–2871, 2015.

P. Ondruska and I. Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. In *Proc. AAAI*, 2016.

R. Riochet et al. IntPhys: A Framework and Benchmark for Visual Intuitive Physics Reasoning. *ArXiv e-prints*, 2018.

O. Ronneberger and T. Fischer, P.and Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015.

A. Sanchez-Gonzalez et al. Graph networks as learnable physics engines for inference and control. *Proceedings of the 35th International Conference on International Conference on Machine Learning*, 2018.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

R. Stewart and S. Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *AAAI*, pp. 2576–2582, 2017.

J. Tompson et al. Accelerating Eulerian Fluid Simulation With Convolutional Networks. *ArXiv e-print arXiv:1607.03597*, 2016.

S. van Steenkiste et al. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *Proc. ICLR*, 2018.

Z. Wang et al. Neural allocentric intuitive physics prediction from real videos. *arXiv preprint arXiv:1809.03330*, 2018.

N. Watters et al. Visual interaction networks: Learning a physics simulator from video. In *Proc. NeurIPS*, 2017.

J. Wu et al. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Proc. NeurIPS*, pp. 127–135, 2015.

J. Wu et al. Physics 101: Learning physical object properties from unlabeled videos. In *Proc. BMVC*, 2016.

J. Wu et al. Learning to see physics via visual de-animation. In *Proc. NeurIPS*, 2017.

A1 DATA GENERATION

In fig. A1 and fig. A2 we provide more details on the scene aspect. For solid background color, the objects and background colours are sampled to be different. Every rectangular object have height and width randomly sample in between 10 and 17 pixels. Custom object are loaded from template images and randomly scaled from 1 to 2.

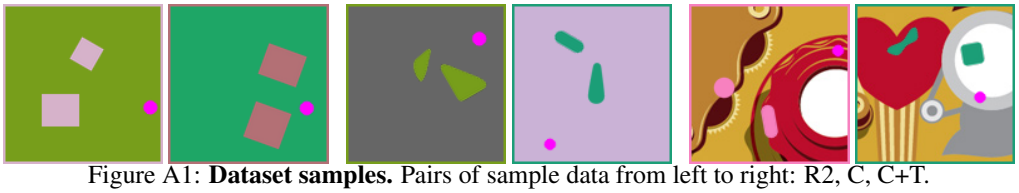


Figure A1: **Dataset samples.** Pairs of sample data from left to right: R2, C, C+T.

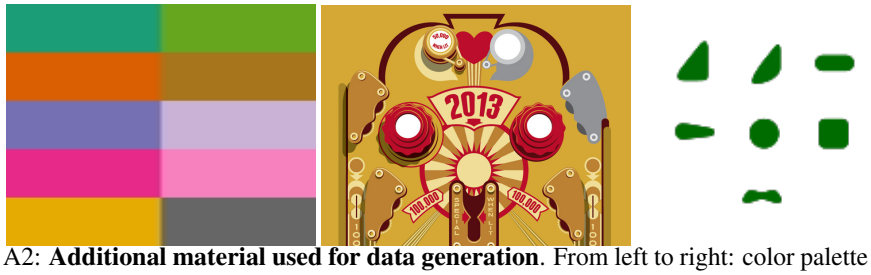


Figure A2: **Additional material used for data generation**. From left to right: color palette for solid object/background color (R2,R4,C), texture used to increase background complexity in (C+T), set of custom objects for (C) data generation.

A2 IMPLEMENTATION DETAILS

A2.1 NETWORK ARCHITECTURE

In fig. A3 we show the details of all the networks of our pipeline. The regression network mentionned in sec.5 is a simple 3 layers fully convolutional network with [3x3] kernel convolution and intermediate channel of size 64.

A2.2 TRAINING DETAILS

In the overall loss l is:

$$\ell(t) = \lambda_{\mathbf{I}} \|\mathbf{I}_t - \hat{\mathbf{I}}_t\|^2 + \lambda_{\mathbf{x}} \|\mathbf{x}_t - \eta(\mathbf{I}_t(\mathcal{R}))\|^2 + \lambda_p \|e(\mathbf{I}_t) - e(\hat{\mathbf{I}}_t)\|^2$$

With t ranging from 0 to T_{train}. For $t = \{0, .., T_0\}$ we used $\lambda_{\mathbf{I}} = 1$ and $\lambda_{\mathbf{x}} = \lambda_{\mathbf{p}} = 0$. In such case $\hat{I}_t = g(x_t, \mathbf{a}, \hat{\mathbf{I}}) = g(\eta(\mathbf{I}_t), \mathbf{a}, \hat{\mathbf{I}})$.

For $t > T_0$, when training with L^2 only we used $\lambda_{\mathbf{I}} = \lambda_{\mathbf{x}} = 1$ and $\lambda_p = 0$. When finetuning with perceptual we weighted the perceputal loss with coefficient $\lambda_p = 10$ and the other losses with $\lambda_{\mathbf{I}} = \lambda_{\mathbf{x}} = 0.01$. Finally, perceptual loss uses feature extracted from conv3 of VGG-16.

The Interaction Network baseline is a simple Interaction Network to which state we concatenated the background feature extracted from a pre-trained VGG-16 network. The state propagator uses the last 4 states. The network is then trained over $T_{train} = 40$ time steps with Adam optimizer with learning rate 10^{-4} and batch size 50. We found that 50 000 iterations were sufficient to reach convergence.

All experiments are run using single NVIDIA-GPU Titan X.

A2.3 EVALUATION DETAILS

To detect blob create a binary image by manually thresholding the intermediate heatmap $\mathbf{x}_t$ and using a simple Hough circle detector. Every threshold coefficient is hand picked for every model but kept constant across scenarios and different data.

For position errors we used an L_2 loss, when more than one object was detected we took the distance to the closest detected object and applied a fix penalty to the loss for all other detection (half the board size).

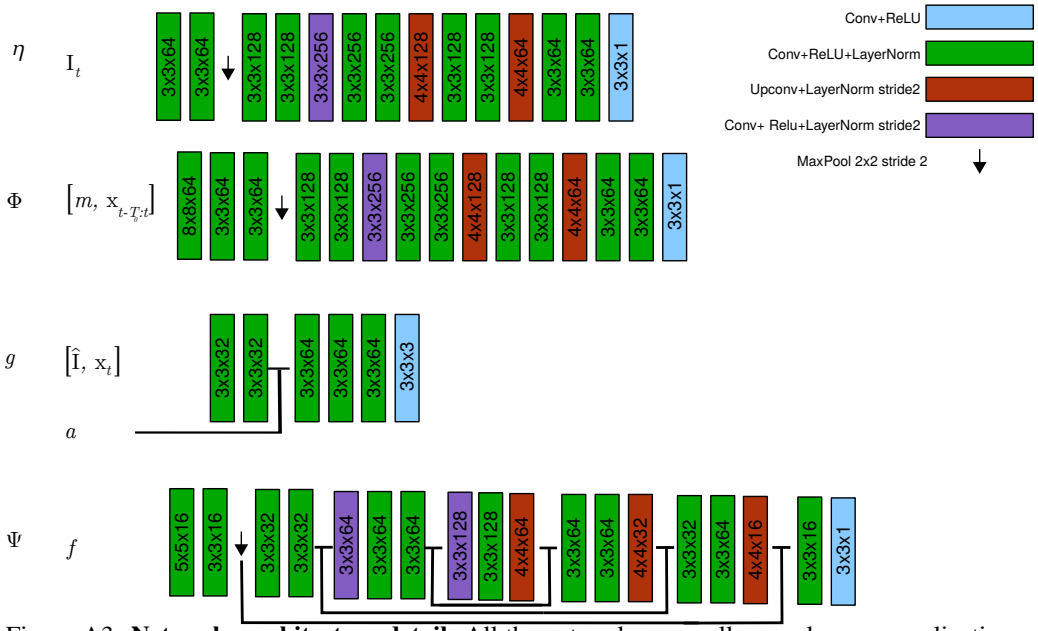


Figure A3: **Networks architecture detail**. All the networks generally uses layer normalization η and Φ are simple auto-encoder type architecture while Ψ shares similarity with U-Net.

Table A1: **Predicting one moving object.** Obstacle type is R2. The test board size is 64×64 We test the average prediction error at $T_{\text{test}} = 20, 60, 100$ well above the duration $T_{\text{train}} = 20$ observed during training. Position errors were normalized by the board size diagonal.

			$T_{\text{test}} = T_{\text{train}} = 20$			$T_{\text{test}} = 3 \times T_{\text{train}}$			$T_{\text{test}} = 5 \times T_{\text{train}}$		
No.	Sup.	Train loss	# obj.	Vid. L_2	Pos. err.	# obj.	Vid. L_2	Pos. err.	# obj.	Vid. L_2	Pos. err.
(1)	Oracle	Perc.	1.0±0.1	2.5±2.9	.013±.012	1 .0±0.1	5.8±4.4	.095±.095	1.0±0.3	6.2±4.5	.201±.136
(2)	None	Perc.	1.0±0.1	3.2±3.1	.030±.068	1.0±0.2	5.5±4.0	.136±.151	0.9±0.4	5.5±4.0	.262±.180

A3 Additional results

In Table A1 we report two more results on one moving object prediction. (1) is an upper bound of our system had full knowledge of $\{B, U\}$ obstacle locations at training time. We simply replaced $\boldsymbol{m}$ with the binary map of B obstacles and $\boldsymbol{a}$ with the binary map of U obstacles where 1s indicate the presence of the obstacle type. In this scenario we note although that we obtain an upper bound of the results introduced in Table 1, our method still remains competitive with this upper bound. (2) reports results of the network trained on C+T and evaluated back on R2. As in Table 3 we see that the network still manage to make accurate predictions. Table A2 is an extension of Table 2 in the paper for the **supervised model** and Table A3 introduces results on multiple balls prediction.

Table A2: **Importance of experience.** For the **supervised** model, we report the trajectory prediction error at $T = 40$ (rows 1-3) and the obstacle map prediction error (rows 4-6) for different obstacle types. The number of runs N in the experiences is varied from 0 to 50. Position errors were normalized by the board size diagonal. $T_{\text{train}} = 20$.

No	Err.	Obst.	$N = 0$	1	2	5	7	10	20	50
(1)	Pos.	R2	1.81±1.41	.94±.975	.53±.81	.15±0.41	.08±.21	.07±.16	.10±.24	.19±.38
(2)	Pos.	R4	3.28±1.94	2.22±1.67	1.56±1.40	.63±.84	.43±.68	.37±.60	.44±.61	.83±.97
(3)	Pos.	C	1.05±0.91	.58±.61	.36±.49	.15±.29	.13±.26	.12±.22	.16±.26	.24±.36
(4)	Obs.	R2	4.1±4.0	3.6±3.8	3.1±3.4	2.3±2.5	2.3±2.4	2.3±2.4	2.3±2.7	2.6±3.0
(5)	Obs.	R4	6.1±2.2	4.1±3.8	3.1±3.4	2.8±3.3	2.7±3.1	2.6±3.0	2.7±2.9	3.3±3.6
(6)	Obs.	C	3.9±3.9	3.3±3.7	2.8±3.1	2.4±2.7	2.4±2.8	2.3±2.6	2.4±2.6	2.5±2.9

Table A3: **Predicting multiple moving object.** Obstacle type is R2, all network were trained on the fully unsupervised with perceptual loss. Num. is the maxmium number of balls per run during training. For instance Num=3 means that every runs sampled contained one to three balls. The test board is size 64×64 We test the average prediction error at $T_{\text{test}} = 20, 60, 100$ well above the duration $T_{\text{train}} = 20$ observed during training.

	$T_{\text{test}} = T_{\text{train}} = 20$		$T_{\text{test}} = 3 \times T_{\text{train}}$		$T_{\text{test}} = 5 \times T_{\text{train}}$	
Num.	# obj.	Vid. L_2	# obj.	Vid. L_2	# obj.	Vid. L_2
			3 Balls			
1	2.6±0.8	6.5±5.2	1.7±0.9	8.5±5.9	1.6±0.8	8.5±6.0
3	3.1±0.6	4.6±4.8	2.9±0.8	10.4±7.6	2.9±0.9	11.0±8.0
			5 Balls			
1	3.3±1.0	8.9±6.6	2.0±1.0	10.2±7.4	3.1±0.6	1.8±.775
3	5.5±0.7	6.6±6.2	5.1±1.1	13.0±9.5	5.1±1.0	13.5±10.0

Chapter 6

RELATE: physically plausible multi-object scene synthesis using structured latent spaces

To conclude this first part of the thesis we finally address the challenge of physical plausibility. While in chapter 5 we proposed a generative method to map environment properties from past experiences, no specific loss enforced plausibility. However, plausibility is central when making predictions about physical scenarios as one can easily tell whether a sequence or a scene is violating the laws of physics. Besides it should further improve the image quality in long term prediction as most objects retain their physical properties over time.

In this chapter, we propose a generative architecture which overcomes the limitations of previous *object-centric* generative models by proposing a correlation module inspired by neural physics (Chang et al., 2017b). This module takes into account relations between objects and between objects and their environment in order to maximise the physical plausibility of a generated scene. Physical plausibility is then enforced by a discriminator implemented as a neural network. We show that this method can work on both static and dynamic scenarios and it gets *state-of-the-art* results on image generation compared to other object-centric methods.

This work was presented at the Conference on Neural Information Processing Systems, 2020.

RELATE: Physically Plausible Multi-Object Scene Synthesis Using Structured Latent Spaces

Sébastien Ehrhardt [1]* **Oliver Groth**[1]* **Aron Monszpart**[3] **Martin Engelcke**[1]
Ingmar Posner[1] **Niloy Mitra**[2] **Andrea Vedaldi**[1]
[1]Department of Engineering Science, University of Oxford
[2]Department of Computer Science, University College London
[3]Niantic
{hyenal,ogroth}@robots.ox.ac.uk

Abstract

We present RELATE, a model that learns to generate physically plausible scenes and videos of multiple interacting objects. Similar to other generative approaches, RELATE is trained end-to-end on raw, unlabeled data. RELATE combines an object-centric GAN formulation with a model that explicitly accounts for correlations between individual objects. This allows the model to generate realistic scenes and videos from a physically-interpretable parameterization. Furthermore, we show that modeling the object correlation is *necessary* to learn to disentangle object positions and identity. We find that RELATE is also amenable to physically realistic scene editing and that it significantly outperforms prior art in object-centric *scene* generation in both synthetic (CLEVR, ShapeStacks) and real-world data (street traffic scenes). In addition, in contrast to *state-of-the-art* methods in object-centric generative modeling, RELATE also extends naturally to dynamic scenes and generates *videos* of high visual fidelity[2].

1 Introduction

We consider the problem of learning to generate plausible images of scenes starting from parameters that are physically interpretable. Furthermore, we wish to learn such a capability from raw images alone, without any manual or external supervision. Image generation is often approached via Generative Adversarial Networks (GAN) [10]. These models learn to map noise vectors, used as a source of randomness, to image samples. While the resulting images are realistic, the random vectors that parameterize them are not interpretable. To address this issue, authors have recently proposed to *structure* the latent space of deep generative models, giving it a partial physical interpretability [28, 29, 36]. For example, HoloGAN [28] samples volumes and cameras to generate 2D images of 3D objects, and BlockGAN [29] creates scenes by composing multiple objects. The resulting GANs are shown to learn concepts such as viewpoint and object disentangling from raw images.

BlockGAN is of particular interest because, via its relatively strong architectural biases, it provides *interpretable* parameters for the scene, incorporating concepts such as position and orientation. However, BlockGAN comes with a significant limitation in that it assumes that objects are mutually *independent*. This approximation is acceptable only when objects interact weakly, but it is badly violated for medium to densely packed scenes, or for scenes such as stacking wooden blocks or cars following a path, where the (object) correlation is strong.

*indicates equal contribution
[2]See more results at `http://geometry.cs.ucl.ac.uk/projects/2020/relate/`.

Preprint. Under review.

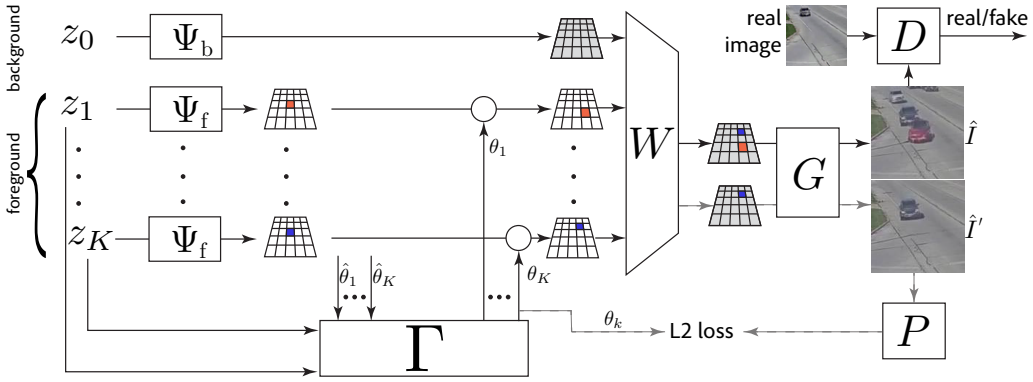


Figure 1: Image generation using RELATE. Individual scene components like background and foreground objects are represented by appearance z_0 and pairs of appearance and pose vectors $(z_i, \theta_i), i \in \{1, \ldots, K\}$, respectively. The key spatial relationship module Γ adjusts the initial independent pose samples $\hat{\theta}_i$ to be physically plausible (*e.g.*, non-intersecting) to produce θ_i. The structured scene tensor W is finally transformed by the the generator network G to produce an image $\hat{I}$. RELATE is trained end-to-end in a GAN setup (D denotes the discriminator and P the position regressor) on real unlabelled images.

Recent work in object-centric generative modeling has attempted to specifically address this by capturing correlations in latent space (*e.g.*, [7, 36]). However as object state information remains significantly entangled in these models they have, to date, been unable to operate on real-world data.

In this paper, we introduce RELATE, a model which explicitly leverages the strong architectural biases of BlockGAN to effectively model correlations between latent object state variables. This leads to a powerful model class, which is able to capture complex physical interactions, while still being able to learn from raw visual inputs alone. Empirically, we show that only when we model such interactions our GAN model correctly disentangles different objects when they exhibit even a moderate amount of correlation (figures 2 and 3). Without this component, the model may still generate high fidelity images, but it generally fails to establish a physically-plausible association between the parameters and the generated images. Our results also demonstrate that GANs are surprisingly sensitive to the correlation of objects in natural scenes, and can thus be used to directly learn these *without* resorting to techniques such as variational auto-encoding (VAE [19]).

We demonstrate the efficacy of RELATE in several scenarios, including balls rolling in bowls of variable shape [6], cluttered tabletops (CLEVR [16]), block stacking (ShapeStacks [12]), and videos of car traffic at a busy intersection. By ablating the interaction module, we show that modeling the spatial correlation between the objects is key. Furthermore, we compare RELATE to several recent GAN- and VAE-based baselines, including BlockGAN [29], GENESIS [7] and OCF [1], in terms of *Fréchet Inception Distance (FID)* [13], and outperform even the best state-of-the-art model by up to 29 points.

Qualitatively, we show that modeling spatial relationships strongly affects scene decomposition and the enforcement of spatial constraints in the generated images. We also show that the physically interpretable latent space learned by RELATE can be used to edit scenes as well as to generate scenes outside the distribution of the training data (*e.g.*, containing more or fewer objects). Finally, we show that the parameterization can be used to generate long plausible video sequences (as measured according to FVD score [34]) by simulating their dynamics while preserving their spatial consistency.

2 Related Work

Interpretable Object-Centric Visual Models. Inspired by the *analysis-by-synthesis* approach for visual perception discussed in cognitive science [44], recent work [3, 7, 11, 36] propose structured latent space models to explain and synthesize images as sets of constituent components which are individually represented using VAEs [19] or GANs [10]. Other approaches favor explicit symbolic representations over distributed ones when parsing an image [33, 40] or propose probabilistic programming languages to formalize image generation [23]. In both cases, object-centric modeling

allows decomposition of images into components and also enables targeted image modification via interpolation in symbol or latent space, *e.g.*, altering position or color of an object parsed by the model. Interpretable and controllable factors of image generation are desirable properties for neural rendering models and have been investigated in recent image generation models, *e.g.*, [1, 25, 28, 29]. However, despite the modeling effort put into the object representations, inter-object interactions are typically only modeled in a less explicit way, *e.g.*, via image layers [36, 40], depth ordering variables [1] or an autoregressive *scene prior* [7]. Our work adds to this line of work by proposing a spatial correlation network which facilitates disentanglement of learned object representations and can be trained from raw observations.

Neural Physics Approximation. Harnessing the power of deep learning to approximate physical processes is an emerging trend in the machine learning community. Especially the approximation of rigid body dynamics with neural networks already boasts a large body of literature, *e.g.*, [2, 4, 8, 22, 35, 39]. Such learned approximations of object interactions have been successfully employed in object manipulation [15] and tracking [9, 21]. However, most entries in this line of work are only applied to visual toy domains or rely on segmentation masks or bounding boxes to initialize their object representations before the networks approximate the object dynamics. While we leverage ideas from neural dynamics modeling, we go beyond the established scope of visual toy domains such as colored point masses or moving MNIST digits [24] and learn directly from rich visual data such as simulated object stacks and real traffic videos without further annotation.

Extraction and Generation of Video Dynamics. Videos are a natural choice of data source to learn about the physics of rigid bodies. Physical information extracted from videos can either be explicit such as estimates of velocity or friction [42] or implicitly represented in the latent space, *e.g.*, sub-spaces corresponding to pose variation [5]. More recently, object-centric approaches have also been leveraged to acquire better video representations for future frame prediction [43] or model-based reinforcement learning [37]. Several studies have also attempted to learn entire video distributions as spatio-temporal tensors in GAN frameworks [17, 32, 38, 41] yielding impressive first results for full video generation in artificial and real domains. In contrast to prior art, our model departs from a monolithic spatio-temporal tensor representation over an entire video. Instead we cast the video learning and generation process as temporal extension of the object-centric representation of a single frame, lowering the computational burden while still faithfully representing long-range dynamics.

3 Method

RELATE (figure 1) consists of two main components: An interaction module, which computes physically plausible inter-object and object-background relationships, and a scene composition and rendering module, which features an interpretable parameter space factored into appearance and position vectors. The details are given next.

3.1 Physically-interpretable scene composition and rendering

RELATE considers scenes containing up to K distinct objects. The model starts by sampling *appearance parameters* $z_1, \dots, z_K \sim \mathcal{U}([-1, 1]^{N_f})$ for each individual foreground object as well as a parameter $z_0 \sim \mathcal{U}([-1, 1]^{N_b})$ for the background. These parameters are small noise vectors, similar to the ones typically used in generative networks. Different from the object poses below, they are sampled independently, thus assuming that the appearance of different objects is independent.

For rendering an image, the appearance parameter z_k is first mapped to a tensor $\Psi_k \in \mathbb{R}^{H \times H \times C}$. This is done via two separate learned decoder networks, one for the background $\Psi_0 = \Psi_b(z_0)$ and one for the foreground objects $\Psi_k = \Psi_f(z_k)$. Here H is the horizontal and vertical spatial resolution of the representation (see table A2) and C is the number of feature channels (see tables A3 and A4). Since we assume that individual objects are much smaller than the overall scene, we restrict Ψ_k, $k \geq 1$ to be non-zero only in a fixed window of size $H' < H$ in the center of the tensor.

Each foreground object also has a corresponding *pose parameter* θ_k, which is geometrically interpretable. For simplicity, we assume $\theta_k \in \mathbb{R}^2$ to be a 2D translation, acting on the tensor Ψ_k via bilinear resampling:

$$\hat{\Psi}_k = \theta_k \cdot \Psi_k \quad \text{such that} \quad [\hat{\Psi}_k]_u = [\Psi_k]_{u+\theta_k}$$

where $u \in \mathbb{R}^2$ is a spatial index and $[\cdot]_u$ means accessing the column of the tensor in bracket at spatial location u (using padding and bilinear interpolation if u does not have integer coordinates). However, θ_k can easily be extended to represent full 3D transformations as previously shown in BlockGAN [29].

Foreground and background objects are composed into an overall scene tensor $W \in \mathbb{R}^{H \times H \times C}$ via element-wise max- (or sum-) pooling as $W_u = \max_{k=0,\dots,K}[\hat{\Psi}_k]_u$. In this manner, the scene tensor is a function $W(\Theta, Z)$ of the pose parameters $\Theta := (\theta_1, \dots, \theta_K)$ and the appearance parameters $Z := (z_0, z_1, \dots, z_K)$. Finally, a decoder network $\hat{I} = G(W)$ renders the composed scene as an image (see table A5).

Discussion. This model is 'physically interpretable' in the sense that it captures (1) the identities of K distinct objects and (2) their pose parameters as translation vectors. This should be contrasted to traditional GAN models, where the code space is given as an uninterpretable, monolithic noise vector z. Despite the structure given to the code space, there is no guarantee that the model will actually learn to map it to the corresponding structure in the example images. However, we found empirically that this is the case as long as the correlations between the different objects are also captured.

3.2 Modeling correlations in scene composition

RELATE departs significantly from prior art such as BlockGAN as it does not assume the parameters θ_i of the different objects to be independent. In order to model correlation, we propose a two-step procedure, based on a residual sampler. First, we sample a vector of K i.i.d. poses $\hat{\Theta} \sim \mathcal{U}([-H''/2, H''/2]^{2K})$ where $H'' < H$ is smaller than the spatial size H of the tensor encoding. Then, we pass this vector to a 'correction' network Γ that remaps the initial configuration to one that accounts for the correlation between object locations and appearances, as well as between objects and the background (coded by the appearance component z_0 in z): $\Theta := \Gamma(\hat{\Theta}, Z)$. In practice, we expect object interactions, as any physical law, to be *symmetric* with respect to the order of the objects. We obtain this effect by implementing Γ as running K copies of the *same* corrective function in parallel:

$$\theta_k = \hat{\theta}_k + \zeta(\hat{\theta}_k, z_k, |z_0, \{z_i, \hat{\theta}_i\}_{i \geq 1, i \neq k}). \tag{1}$$

The function ζ is implemented in a manner similar to the Neural Physics Engine (NPE) [4]:

$$\zeta(\hat{\theta}_k, z_k, |z_0, \{z_i, \hat{\theta}_i\}_{i \geq 1, i \neq k}) = f(\hat{\theta}_k, z_k, z_0, h_k^s), \quad h_k^s = \sum_{q \neq k} g(\hat{\theta}_k, z_k, \hat{\theta}_q, z_q), \tag{2}$$

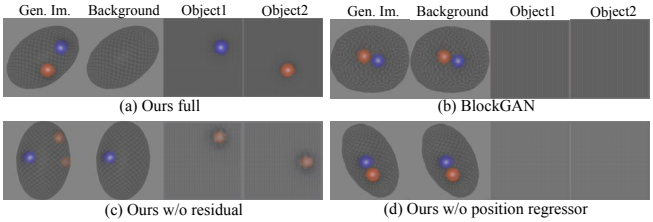
where f and g are Multi Layer Perceptrons (MLPs) (tables A6, A7) operating on stacked vector inputs and h^s is an embedding capturing the interactions between the K objects. Besides symmetry, an advantage of this scheme is that it can take an arbitrary number of objects K due to the sum-pooling operator used to capture the interactions. In this manner, the sampler Γ is automatically defined for any value of K, which is sampled uniformly from an interval $[K_{\text{min}}, K_{\text{max}}]$. Furthermore, sampling independent quantities followed by a correction has the benefit of injecting some variance on the objects' positions at the early stage of training, which helps to avoid converging to trivial/bad solutions.

Ordered scenes. An advantage of RELATE is that it can be easily modified to take advantage of additional structure in the scene. For scenes where objects have natural order, such as stacks of blocks, we experiment with conditioning pose θ_i on the preceding pose θ_{i-1}, using a Markovian process. This is done by first sampling $\hat{\theta}_1 \sim \mathcal{U}([-H''/2, H''/2])$, and then applying a correction to account for the background z_0 as before, finally sampling the other objects in sequence:

$$\theta_1 = \hat{\theta}_1 + f_0(\hat{\theta}_1, z_1, z_0), \qquad \forall k > 1: \quad \theta_k = \theta_{k-1} + f_1(\theta_{k-1}, z_{k-1}, z_0), \tag{3}$$

where f_0, f_1 are implemented as MLPs as before (tables A10, A11). Note that this can be interpreted as a special case of the model above in the sense that we can write $\Theta := \Gamma(\hat{\Theta}, Z)$, provided that $\hat{\theta}_k = 0$ for $k \geq 2$.

Modeling dynamics. RELATE can also be immediately extended to make dynamic predictions. For this, we sample the initial positions $\theta_k(0)$ as before and then update them incrementally as $\theta_k(t+1) = \theta_k(t) + v_k(t+1)$, where $v_k(t)$ is the object velocity. In order to obtain the latter, we let $V_k(t) = [v_k(t-i)]_{i=2,1,0}$ denote the last three velocities of the k-th object. The initial



(a) Ours full (b) BlockGAN

(c) Ours w/o residual (d) Ours w/o position regressor

Figure 2: Ablation Study. For every case we render every component of our method independently. We show that only our full model is able to correctly disentangle individual components of the scene.

Table 1: Ablation study. FID score (lower is better) on BALLSINBOWL. Ours (full) reaches the highest fidelity by a large margin.

BlockGAN 2D[29]	149.8
Ours w/o residual	137.1
Ours w/o pos. reg.	147.8
Ours (full)	**77.4**

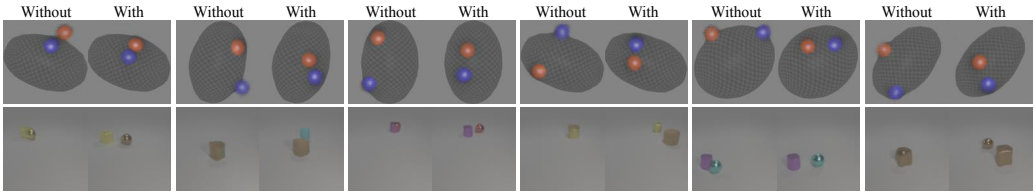


Figure 3: Effect of the interaction module Γ. We show pairs of images without and with the correction function Γ is applied. For BALLSINBOWL the correction moves the balls within the bowl, and for CLEVR it pushes apart intersecting objects.

value $V_k(0) = e_v(z_k, z_0, \theta_k(0))$ is initialized as a function of the appearance parameters and initial positions (table A8); and we use the NPE style update equations [4], where e_v, f_v and g_v are MLPs,

$$v_k(t+1) = f_v(\theta_k(t), z_k, V_k(t), z_0, h_k^d(t)), \quad h_k^d(t) = \sum_{q \neq k} g_v(\theta_k(t), z_k, V_k(t), \theta_q(t), z_q, V_q(t)). \quad (4)$$

3.3 Learning objective

Training our model makes use of a training set I_i, $i = 1, \ldots, N$ of N images of scenes containing different object configurations. No other supervision is required. Our learning objective is a sum of *two high fidelity losses* and *a structural loss* which we describe below.

For high fidelity, images $\hat{I}$ generated by the model above are contrasted to real images I from the training set using the standard GAN discriminator $\mathcal{L}_{\text{GAN}}(\hat{I}, I)$ and style $\mathcal{L}_{\text{style}}(\hat{I}, I)$ losses from [29].

In addition, we introduce a regularizer to encourage the model to learn a non-trivial relationship between object positions and generated images. For this, we train a position regressor network P that, given a generated image $\hat{I}$, predicts the location of the objects in it. In practice, we simplify this task and generate an image $\hat{I}'$ by retaining only object k of the K objects at random and minimizing $\|\breve{\theta}_k - P(G(W(z_0, z_k, \theta_k)))\|_2^2$. Here the symbol $\breve{}$ means that gradients are not back-propagated through θ_k: this is to avoid mode collapse of the position at zero. P shares most of its weights with the discriminator network (see table A9).

In the case of dynamic prediction, the discriminator takes as input the sequence of images concatenated along the RGB dimension and is tasked to discriminate between fake and real sequences. Similar to a static model we also have a position regressor which is tasked to predict the position of an object rendered at random with zero velocity.

4 Experiments

Implementation details. We learn mappings Ψ_b and Ψ_f using the same Adaptive Instance Normalization (AdaIN) [14] architecture. The spatial size of their output tensors is set to $H = 16$ and the final output image to 128×128 (which is reduced when needed for fair comparison to other methods). We used the Adam [18] optimizer for learning and train for a fixed number of epochs and always select the last model snapshot. We consider GENESIS [7] and OCF [1] baselines, quoting results from the original papers whenever possible, and BlockGAN2D as an ablation of our method.

Table 2: Comparison to state-of-the-art methods. FID score (lower is better) for various datasets. We consistently outperfom prior art in object centric scene generation. 'Ordered' refers to the variant discussed in sec. 3.2. Note that on ShapeStacks, BlockGAN2D collapses to render all towers in the background and is not object-centric therefore (see fig. A3). * are models which FID score are evaluated on 64×64 images for 'General' variant (see appendix A3)

RELATE variant	CLEVR-5 General	CLEVR-5vbg General	CLEVR General	ShapeStacks Ordered	REALTRAFFIC General
DCGAN*[30]	264.8	361.8	247.8	197.6	47.6
DRAGAN*[20]	80.8	84.4	108.0	57.2	38.8
OCF [1]	N/A	83.1	N/A	N/A	N/A
MultiGAN[36]	88.2	71.4	87.2	N/A	47.6
GENESIS [7]	211.7	169.4	151.3	233.0	167.1
BlockGAN2D [29]	73.2	63.8	94.4	105.1	57.9
Ours	**70.5**	**54.3**	**79.7**	**102.3**	**42.0**

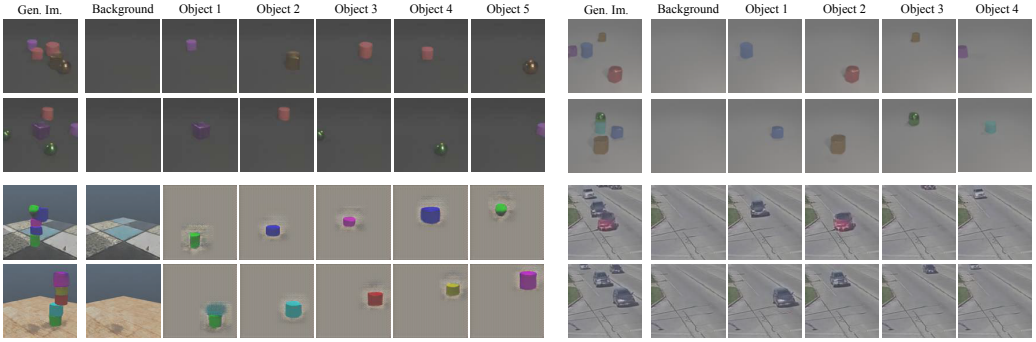


Figure 4: Component-wise scene generation. From a generated image (left) RELATE can render each component individually for each dataset. For CLEVR and REALTRAFFIC objects are rendered after being composed with the background (cf. section 4.2). Top left picture has increased contrast for easier visualization.

Datasets. We conduct experiments on four different datasets. First, we consider a relatively simple dataset, BALLSINBOWL from [6], for assessing the model features and ablations. This data consists of videos of two distinctly colored balls rolling in an elliptical bowl of variable orientation and eccentricity. Interactions amount to object collisions and the fact that they must roll within the bowl. To this, we add two popular synthetic datasets CLEVR [16] (cluttered tabletops) and ShapeStacks [12] (block stacking). Finally, we collected a new dataset REALTRAFFIC containing five hours of footage of a busy street intersection, divided into fragments containing from one to six cars. Especially the last dataset contains many interactions between the individual cars as they adapt their speed to the surrounding traffic which happens frequently when the light changes and cars either slow down because of a queue on red or accelerate when the lights change to green again. Further details about training, evaluation and datasets can be found in the appendix sections A3, A4 and A6.

4.1 Generating static scenes

Ablation study. We start experimenting with the comparatively simple BALLSINBOWL dataset to conduct basic ablations. The first ablation removes the spatial correlation module Γ and the position regression loss, therefore reducing RELATE to a 2D version of BlockGAN. We also consider *'w/o residual'*, where the addition $\hat{\theta}_k$ in equation (1) is removed, and *'w/o pos. loss'*, where the position regression loss regularizer is removed. Table 1 shows that each component of RELATE yields an improvement in terms of FID scores on this dataset supporting our spatial modeling decisions. Furthermore, in figure 2 we show qualitatively that only RELATE (a) is able to correctly disentangle the underlying scene factors. We do this by generating the same image while retaining a single factor, which correctly isolates the background, and, in turn, both individual objects. BlockGAN 2D (b) and *'ours w/o pos. reg.'* (d) fail to disentangle the factors entirely, mapping everything to the background component. *'Ours w/o residual'* (c) shows that the model partially fails to disentangle,

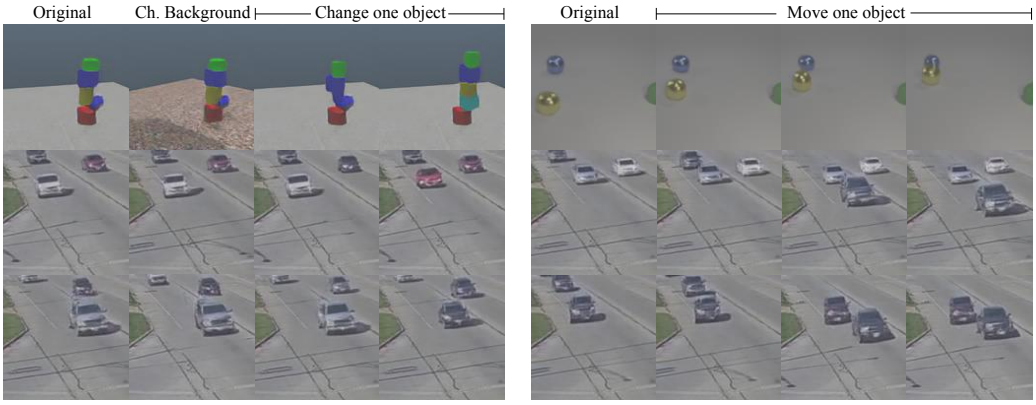


Figure 5: Image editing. Left: We demonstrate the capacity of RELATE to change the background and the appearance of individual objects. Right: RELATE is also able to modify the position of a single object.

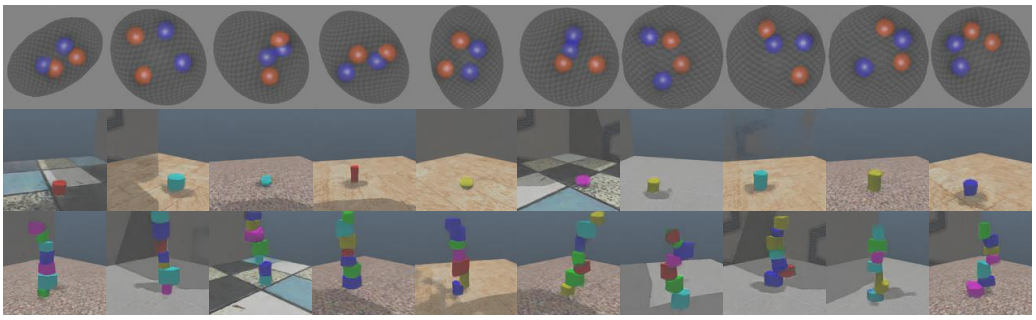


Figure 6: Out-of distribution generation. RELATE can generate images outside the training distribution. The first row shows generating a bowl with four balls, whereas the training set only features exactly two. The last two rows depict towers of one and seven objects, whereas the training images only had stacks of height two to five.

with the background encoding some but not all the objects. Finally, figure 3 visualizes the effect of the interaction module Γ. Recall that this is implemented as a 'correction' function that accounts for correlation starting from independently-sampled parameters. For BALLSINBOWL, the correction module moves the balls within the bowl, and for the CLEVR it pushes objects apart if they intersect.

Quantitative evaluation. In table 2, we compare RELATE to existing scene component-wise generators (row 3-6) on ShapeStacks, CLEVR and REALTRAFFIC and also simple GAN baselines such as DCGAN[30] or DRAGAN[20] (row 1-2). We report performance in terms of FID [13] score between 10,000 images sampled from our model and the respective test sets. For CLEVR, we train RELATE and BlockGAN on a restricted version of the data containing from three to six objects in an image[3], and at test time, we require all models to sample images with three to ten objects. We consistently outperform all prior component-wise methods in all scenes and scenarios according to FID scores. Only in ShapeStacks we are outperformed by DRAGAN because of the quality of the background generated by this method. In particular, on CLEVR, RELATE can generate a larger number of objects than seen during training suggesting its improved generalization capabilities which are demonstrated further in figure 6.

4.2 Interpretability of the latent space and scene editing

As shown in the ablation studies in figure 2, RELATE successfully disentangles a scene into independent components - in contrast to BlockGAN2D which struggles to separate individual objects from the background. Figure 4 shows that RELATE can *disentangle* also far more complex scenes in REALTRAFFIC, ShapeStacks and CLEVR. Note that for REALTRAFFIC and CLEVR we render objects composed with the background. In fact in these datasets the size and appearance of each

[3]Note that GENESIS was trained on the full training set featuring three to ten objects.

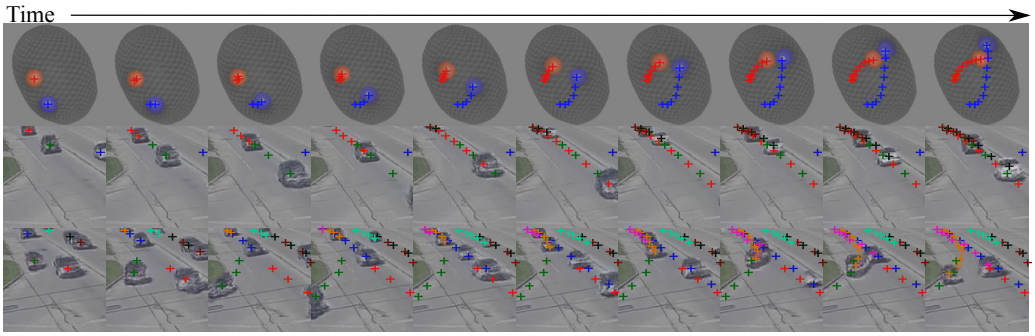


Figure 7: Video generation. We show consecutive video frames generated by RELATE overlayed with crosses representing projections of the model's estimated pose parameters for each object. In BALLSINBOWL the interaction with the environment is well captured as the balls stay within the bowl. In REALTRAFFIC the cars stay in their lane, or can decide to make a right turn (last row).

Figure 8: Failure case. Our model struggles to understand big changes of aspect ratio. For instance, in the static case, when we drag two cars towards the bottom of the image we can see that the left grey car disappears at some point (highlighted in red) and reappears on the edge of the image. This explains why the video generation on REALTRAFFIC fails to capture the true data distribution more faithfully resulting in lower FVD score.

object is correlated to their position in the background because of the camera perspective. Next, in figure 5 we use RELATE to *edit* a generated scene. For example we can change the position or identity of individual objects. Finally, we show that RELATE can generate *out-of-distribution* scenes. This is achieved in particular by sampling a different number of objects. In figure 6, for instance, RELATE is trained on ShapeStacks seeing towers of height two to five. However, it can render taller towers of up to seven objects, or even just a single object. Likewise, in BALLSINBOWL it can generate bowls with four balls having seen only two during training. Furthermore, in ShapeStacks each tower is composed of blocks of *different* colors, but RELATE can relax this constraint rendering objects with repeated colors.

4.3 Simulating dynamics

We train this model on BALLSINBOWL and REALTRAFFIC to predict 15 and 10 consecutive frames respectively. During generation, we sample videos with a sequence length of 30 frames and measure the faithfulness with respect to the distribution of the test data via the *Fréchet Video Distance (FVD)* [34]. We achieve FVD scores of 556 and 2253 respectively. This is perceptibly better than 920 and 3370 for a baseline consisting of time-shuffled sequences from the respective training sets, which feature perfect resolution but poor dynamics. Qualitatively in figure 7 we see that the model does understand the motion and captures interaction with the background. For instance in BALLSINBOWL the balls do have a curved motion because of the shape of the bowl and decrease in speed when reaching the edges of the bowl which are in higher position (see first row of figure 7). In REALTRAFFIC the cars do stay in their respective lane. Interestingly our model is able to handle different types of motions correctly (see third row figure 7) and use the sample vector to decide whether the cars should go straight or turn. Finally we see that we can also generate videos with much more cars than the upper bound (5) with which the system was trained (see last row figure 7).

Limitations While the dynamics in video generation look realistic in most cases, REALTRAFFIC also exposed the limitations of our approach. In this dataset the perspective range of the camera is important. As a result, cars at the bottom of the image, which appear bigger in the training data, often do not get generated properly in the static case (see highlighted frames in figure 8). We hypothesize that this is also the main reason why the cars' appearance (and hence FVD score) deteriorates in the dynamic scenarios: since the style parameters z_i are fixed to preserve identity, it is not possible for the model to account accurately for the appearance change introduced by large changes of perspective

over the course of a sequence. To verify our hypothesis in appendix A5 we proposed a simple modification of our pipeline that accounts for scale changes in an image. This modification simply allows the network to predict H' for each individual object instead of it being a constant. We found that in general it allows our network to train on datasets with more important range of scales such as CLEVR3 [1] and renders objects at more different scales for CLEVR and REALTRAFFIC (see figure A1). Besides in table A12 we show that this modification almost always results in higher FID scores from our main model as well as a significant boost of 397 points in FVD score for REALTRAFFIC (1855 vs 2253).

5 Conclusion

We have introduced RELATE, a GAN-based model for object-centric scene synthesis featuring a physically interpretable parameter space and explicit modeling of spatial correlations. Our experimental results suggest that spatial correlation modeling plays a pivotal role in disentangling the constituent components of a visual scene. Once trained, RELATE's interpretable latent space can be leveraged for targeted scene editing such as altering object positions and appearances, replacing the background or even inserting novel objects. We demonstrate our model's effectiveness by presenting *state-of-the-art* scene generation results across a variety of simulated and real datasets. Lastly, we show how our model naturally extends to the generation of dynamic scenes being able to generate entire videos from scratch. A main limitation of our current model is its restriction to planar motions which prevents it from representing arbitrarily 3D motions featuring angular rotation more faithfully, most notably highlighted by the experiments for video generation. However, we believe that our work can contribute to future research in that area by providing a scalable spatio-temporal modeling approach which is conveniently trainable on unlabeled data.

6 Acknowledgments

This work is supported by the European Research Council under grants ERC 638009-IDIU, ERC 677195-IDIU, and ERC 335373. The authors acknowledge the use of Hartree Centre resources in this work. The STFC Hartree Centre is a research collaboratory in association with IBM providing High Performance Computing platforms funded by the UK's investment in e-Infrastructure. The authors also acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility in carrying out this work (`http://dx.doi.org/10.5281/zenodo.22558`). The authors also thank Olivia Wiles for feedback on the paper draft and Thu Nguyen-Phuoc for providing the implementation of BlockGAN.

Broader Impact

Our method advances the ability of computers to learn to understand environments in images in an object-centric way. It also enhances the capabilities of generative models to generate realistic images of "invented" environment configurations.

Overall, we believe our research to be at low to no risk of direct misuse. At present, our generation results are insufficient to fool a human observer. However, it has to be noted that the sampling process is, as in many other deep generative models, capable of revealing patterns observed in the training data, *e.g.*, specific textures or object geometries. Such data privacy concerns are not applicable in the street traffic data used in our research, since the resolution of the videos is far too low to identify individual drivers or recognize cars' license plates. However, 'training data leakage' should be taken into consideration when the model is trained on more sensitive datasets.

In a positive prospect, we believe that our model contributes to further the development of less opaque machine learning models. The explicit object-centric modelling of image components and their geometric relationships is in many of its aspects intelligble to a human user. This facilitates debugging and interpreting the model's behaviour and can help to establish trust towards the model when employed in larger application pipelines.

However, the key value of our paper is in the methodological advances. It is conceivable that, like any advance in machine learning, our contributions could ultimately lead to methods that in turn can

and are misused. However, there is nothing to indicate that our contributions facilitate misuse in any direct way; in particular, they seem extremely unlikely to be misused directly.

References

[1] Titas Anciukevicius, Christoph H Lampert, and Paul Henderson. Object-centric image generation with factored depths, locations, and appearances. *arXiv preprint arXiv:2004.00642*, 2020.

[2] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.

[3] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[4] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations*, 2017.

[5] Emily L Denton and vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4414–4423. Curran Associates, Inc., 2017.

[6] Sebastien Ehrhardt, Aron Monszpart, Niloy J Mitra, and Andrea Vedaldi. Taking visual motion prediction to new heightfields. *Computer Vision and Image Understanding*, 181:14–25, 2019.

[7] Martin Engelcke, Adam R Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations*, 2020.

[8] Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. In *International Conference on Learning Representations*, 2016.

[9] Fabian B Fuchs, Adam R Kosiorek, Li Sun, Oiwi Parker Jones, and Ingmar Posner. End-to-end recurrent multi-object tracking and trajectory prediction with relational reasoning. *arXiv preprint arXiv:1907.12887*, 2019.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

[11] Klaus Greff, Raphaël Lopez Kaufmann, Rishab Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[12] Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 702–717, 2018.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[15] Michael Janner, Sergey Levine, William T. Freeman, Joshua B. Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-oriented prediction and planning. In *International Conference on Learning Representations*, 2019.

[16] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary

visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017.

[17] Nal Kalchbrenner, Aäron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1771–1779. JMLR.org, 2017.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[20] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[21] Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in Neural Information Processing Systems*, pages 8606–8616, 2018.

[22] Jannik Kossen, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting. Structured object-aware physics prediction for video modeling and planning. In *International Conference on Learning Representations*, 2020.

[23] Tejas D Kulkarni, Pushmeet Kohli, Joshua B Tenenbaum, and Vikash Mansinghka. Picture: A probabilistic programming language for scene perception. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 4390–4399, 2015.

[24] Yann LeCun. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

[25] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. *arXiv preprint arXiv:1912.05237*, 2019.

[26] Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the fairness of disentangled representations. In *Advances in Neural Information Processing Systems*, pages 14611–14624, 2019.

[27] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[28] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. HoloGAN: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7588–7597, 2019.

[29] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. BlockGAN: Learning 3d object-aware scene representations from unlabelled images. *arXiv preprint arXiv:2002.08988*, 2020.

[30] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[31] Danilo J Rezende and Fabio Viola. Generalized elbo with constrained optimization, geco. In *Workshop on Bayesian Deep Learning, NeurIPS*, 2018.

[32] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. Temporal generative adversarial nets with singular value clipping. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[33] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017.

[34] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[35] Sjoerd Van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations*, 2018.

[36] Sjoerd van Steenkiste, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly. Investigating object compositionality in generative adversarial networks. *CoRR*, abs/1810.10340, 2019.

[37] Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua B Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. In *International Conference on Learning Representations*, 2019.

[38] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances in neural information processing systems*, pages 613–621, 2016.

[39] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *Advances in neural information processing systems*, pages 4539–4547, 2017.

[40] Jiajun Wu, Joshua B Tenenbaum, and Pushmeet Kohli. Neural scene de-rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 699–707, 2017.

[41] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 91–99. Curran Associates, Inc., 2016.

[42] Tian Ye, Xiaolong Wang, James Davidson, and Abhinav Gupta. Interpretable intuitive physics model. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 87–102, 2018.

[43] Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani. Compositional video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10353–10362, 2019.

[44] Ilker Yildirim, Max Siegel, and Joshua Tenenbaum. 34 physical object representations. *The Cognitive Neurosciences*, pages 399–409, 2020.

Supplementary Material for "RELATE: Physically plausible Multi-Object Scene Synthesis Using Structured Latent Spaces"

In this supplementary material we provide further details about RELATE. We organized this manuscript as follow. We first give additional object decomposition score in appendix A1. Then we describe the losses we use in more details in appendix A2. We follow with a more in-depth description of implementation details in appendix A3. After, appendix A4 is dedicated to explain in more details how the various baselines were trained. Appendix A5 explains in more details the scale augmented model reference in section 4. Appendix A6 contains a more thorough explanation of every dataset and data collection when applicable. Finally we provide more qualitative results in appendix A7.

A1 Additional experiments

Table A1: Disentanglement score. For each dataset of table 2 we report respectively the distance and correlation score described in appendix A1. Our model outperform BlockGAN2D in the most complex scenarios: ShapeStacks. Both model reach similar scores for the other scenes.

RELATE variant	CLEVR-5 General	CLEVR-5vbg General	CLEVR General	ShapeStacks Ordered	REALTRAFFIC General
BlockGAN2D [29]	19.0/0.78	18.0/0.78	18.0/0.79	272.0/0.00	22.0/0.64
Ours	17.0/0.79	17.0/0.84	19.0/0.78	17.0/0.58	23.0/0.63

To provide more quantitative insights of the disentanglement capabilities of our model we ran additional experiments. While measures such as MIG[26] can be used, computing this score is not applicable in our case since our model does not feature an inference component to compute the posterior $q(z|x)$. As an alternative, we toggle each object individually (out of 5 objects generated), looking at how the generated image changes. We then report the distance between the pixel location corresponding to the maximum image change and the location (scaled θ_i) of the object that was toggled. In table A1 we report the median distance as well as the R^2 correlation score for a simple linear regression between θ_i and the pixel location corresponding to the maximum image change. We note that in general that our model outperforms BlockGAN2D, most notably for ShapeStacks where BlockGAN isn't disentangling different objects at all.

A2 Losses

Our final loss is the sum of three losses mentioned in the main text:

$$\mathcal{L}_{tot} = \mathcal{L}_{\text{GAN}}(\hat{I}, I) + \mathcal{L}_{\text{style}}(\hat{I}, I) + \min_{G,\Gamma,P} \|\breve{\theta}_k - P(G(W(z_0, z_k, \theta_k)))\|_2^2,$$

where $\mathcal{L}_{\text{GAN}}(\hat{I}, I)$ is the standard GAN loss:

$$\mathcal{L}_{\text{GAN}}(\hat{I}, I) = \min_{G,\Gamma} \max_{D} \mathbb{E}[\log(1 - D(G(W(Z, \Theta))))] + \mathbb{E}[\log(D(I))]$$

The style loss follows the implementation of BlockGAN [29]. The input of the style discriminator D_l are mean μ_l and variance σ_l^2 across spatial dimensions of $\Phi_l(x) \in \mathcal{R}^{W_l \times H_l \times C_l}$, the output of the lth layer of D taken before the normalization step:

$$\mu_l(\Phi_l(x)) = \frac{1}{W_l \times H_l} \sum_i \sum_j \Phi_l(x)_{i,j},$$

$$\sigma_l^2(\Phi_l(x)) = \frac{1}{W_l \times H_l} \sum_i \sum_j (\Phi_l(x)_{i,j} - \mu_l(\Phi_l(x)))^2.$$

The style discriminator D_l for each layer is then implemented as a linear layer followed by a sigmoid activation function. The resulting style loss is:

$$\mathcal{L}_{\text{style}}(\hat{I}, I) = \max_{D} \sum_l \mathbb{E}[\log(1 - D_l(\hat{I}))] + \mathbb{E}[\log(D_l(I))]$$

Table A2: Hyper-parameters for each datasets. Epoch nums are the number of epochs we trained for.

Dataset	Learning rate	Epoch nums	M	$K_{min} - K_{max}$	H'	N_b	N_f	H''/H sampling range
BALLSINBOWL	0.001	60	1	2-2	8	3	1	$[-0.8, 0.8]^2$
CLEVR5	0.0001	40	2	2-5	4	1	90	$[-0.6, 0.6]^2$
CLEVR5-vbg	0.0001	30	2	2-5	4	1	90	$[-0.6, 0.6]^2$
CLEVR	0.0001	40	2	3-6	4	1	90	$[-0.6, 0.6]^2$
ShapeStacks	0.001	30	2	2-5	4	12	64	$[-0.6, 0.6] \times [0, 0.6]$
REALTRAFFIC	0.0001	20	2	1-5	6	1	20	$[-0.6, 0.6]^2$

A3 Implementation details

Infrastructure and framework For all experiments we use PyTorch 1.4. We train all models on a single NVIDIA Tesla V100 GPU.

Training hyperparameters We initialize all weights (including instance normalization ones) by drawing from a random normal distribution $\mathcal{N}(0, 0.02)$. All biases are initialized to 0. For each update of the discriminator we update the generator M number of times. We use Adam parameters $(\beta_1, \beta_2) = (0., 0.999)$ for all datasets except BALLSINBOWL where $\beta_1 = 0.5$. Similarly W was a max-pooling operator in all datasets except BALLSINBOWL where we used a sum pooling operator. As in BlockGAN[29], background and foreground decoders each start from a learned constant tensors T_b and T_f respectively with sizes $H \times H \times 256$ and $H' \times H' \times 512$. For BALLSINBOWL we use a tensor T_f for each object and use a constant style vector of one.

In the case of dynamic scenarios we reuse same hyperparameters as in the static case except that we use a learning rate of 0.0001 and $\beta_1 = 0$.

Full details of the parameters for each dataset can be found in table A2

Evaluation details For FID scores computation we draw 10 000 samples from our model which we compare against the same number of images drawn from the test set. To compute FVD score on each dataset, we sample 500 videos of 30 frames from our model and compare them against the videos of the respective test sets (500 for BALLSINBOWL and 275 videos for REALTRAFFIC). This also applies to the time shuffled baseline.

To be able to compare with other methods we resize our generated images to 96×96 on CLEVR5 and CLEVR5-vbg and 64×64 for ShapeStacks. For simple baseline DRAGAN and DCGAN we evaluate FID score on the generated 64×64 images from these models. We evaluate on the generated 128×128 images otherwise.

We empirically found that background was rendered with better quality for lower values of z_0. Hence at test time we sampled z_0 from $\mathcal{U}([-0.5, 0.5]^{N_b})$ for optimal background fidelity rendering.

A3.1 Architecture details

Generator. In this work we maintain the core of our architecture fixed as much as possible. Since the dimension of the sample z_i does not necessarily match the channel dimension where it is injected before applying Adaptive Instance Normalisation (AdaIN) to a layer l we map z_i to a vector $\hat{z}_i$ transformed such that

$$\hat{z}_i = \max(W_l^T z_i + b_l, 0)$$

Where (W_l, b_l) are learnable parameters. AdaIN is applied at the end of the layers (after the activation). All LeakyReLU layers are using a parameter of 0.2.

Discriminator We describe the architecture of the discriminator network in more details in table A9. We use spectral normalization [27] at almost every layer. Positions are directly regressed from the last feature output of the discriminator (see last line P_{end}). Therefore in practice P and D share the same backbone D_b (see table table A9 until flatten) for every image I:

$$P(I) = P_{end}(D_b(I)), \qquad D(I) = Disc(D_b(I)).$$

Table A3: Network architecture for the foreground object generator Ψ_f.

Layer name	Layer Type	Input size	Output size	Kernel Size	Stride	Activation	Norm.
Style_f	Id	$H' \times H' \times 512$	$H' \times H' \times 512$	-	-	Id	AdaIn
Convtf_1	ConvTranspose	$H' \times H' \times 512$	$H' \times H' \times 512$	3×3	1	LeakyReLU	AdaIn
Convtf_2	ConvTranspose	$H' \times H' \times 512$	$H' \times H' \times 256$	3×3	1	LeakyReLU	AdaIn
Pad	Padding	$H' \times H' \times 512$	$H \times H \times 256$	-	-	-	-

Table A4: Network architecture for the background object generator Ψ_b.

Layer name	Layer Type	Input size	Output size	Kernel Size	Stride	Activation	Norm.
Style_b	Id	$H \times H \times 256$	$H \times H \times 512$	-	-	Id	AdaIn
Convtb_1	ConvTranspose	$H \times H \times 512$	$H \times H \times 512$	3×3	1	LeakyReLU	AdaIn
Convtb_2	ConvTranspose	$H \times H \times 512$	$H \times H \times 256$	3×3	1	LeakyReLU	AdaIn

Input for style discriminator are taken after the convolution of (Convd_2, Convd_3, Convd_4, Convd_5) in table A9 before the normalization. Spectral Normalization was *not* applied to any D_l.

Table A10: Network architecture for module f_0.

Layer name	Layer Type	Input size	Output size	Activation
FCf_0_1	Linear	$N_f + N_b + 2$	128	LeakyReLU
FCf_0_2	Linear	128	64	LeakyReLU
FCf_0_3	Linear	64	2	Tanh

Table A11: Network architecture for module f_1.

Layer name	Layer Type	Input size	Output size	Activation
FCf_1_1	Linear	$N_f + N_b$	128	LeakyReLU
FCf_1_2	Linear	128	64	LeakyReLU
FCf_1_3	Linear	64	2	None
Pos_{out}	Sigmoid(x) Tanh(y)	2	2	None

A4 Baselines

DCGAN[30] and DRAGAN[20]. We used an online pytorch implementation[4] with default hyperparameters. We trained these models to generate 64×64 images and therefore only evaluated FID score at the same resolution (see appendix A3).

OCF[1]. OCF results were copied from original paper of [1].

MultiGAN[36]. We used official code provided by the authors[5] with their default hyper parameters. We only modifed the generated image size to be similar to ours.

BlockGAN2D. We use the same hyperparameters and network architecture as RELATE except for learning rate and M. In all cases we report the best results over models trained with variations of learning rate in (0.001, 0.0001) and M in (2,3).

GENESIS. We use the official implementation[6] of GENESIS for all experiments. For the ShapeStacks dataset, we use the official model snapshot released with the original paper[7]. For all other datasets, we train GENESIS for 500,000 iterations with the default learning parameters and select the last model checkpoint for evaluation. When training GENESIS we use *constrained ELBO optimization* [31] controlled via `g_goal` in the training script which influences the decomposition capability of GENESIS. We perform a grid search over `g_goal` in the range of 0.5635 to 0.5655 and select the model with the lowest ELBO after 500,000 iterations.

[4] `https://github.com/LynnHo/DCGAN-LSGAN-WGAN-GP-DRAGAN-Pytorch`

[5] `https://github.com/google/compare_gan/tree/v2_multigan/compare_gan/src/multi_gan`

[6] `https://github.com/applied-ai-lab/genesis`

[7] `https://drive.google.com/drive/folders/1uLSV5eV6Iv4BYIyh0R9DUGJT2W6QPDkb?usp=sharing`

Table A5: Network architecture for the generator G. Outputs of all K foreground object generators Ψ_f and background generator Ψ_b are stacked on the first dimension before entering layer W (third row).

Layer name	Layer Type	Input size	Output size	Kernel Size	Stride	Activation
Ψ_f (see table A3)	-	$H' \times H' \times 512$	$16 \times 16 \times 256$	-	-	-
Ψ_b (see table A4)	-	$H \times H \times 512$	$16 \times 16 \times 256$	-	-	-
W	Max/Sum Pool	$(K+1) \times 16 \times 16 \times 256$	$16 \times 16 \times 256$	-	-	-
Convtg_1	ConvTranspose	$16 \times 16 \times 256$	$32 \times 32 \times 128$	4×4	2	LeakyReLU
Convtg_2	ConvTranspose	$32 \times 32 \times 128$	$64 \times 64 \times 64$	4×4	2	LeakyReLU
Convtg_3	ConvTranspose	$64 \times 64 \times 64$	$64 \times 64 \times 64$	3×3	1	LeakyReLU
Convtg_4	ConvTranspose	$64 \times 64 \times 64$	$128 \times 128 \times 64$	4×4	2	LeakyReLU
Convtg_5	ConvTranspose	$128 \times 128 \times 64$	$128 \times 128 \times 3$	3×3	1	Tanh

Table A6: Network architecture for module f and f_v. * indicates modification of f_v

Layer name	Layer Type	Input size	Output size	Activation
FCf_1	Linear	$2 \times (N_f + 2 + 2^*)$	32	LeakyReLU
FCf_2	Linear	32	32	LeakyReLU
FCf_3	Linear	32	32	None

A5 Scale augmented model

To tackle the limitation discussed in section 4.3 we propose to augment our model with scale prediction. Practically this sums up to predict H' for each individual objects. Therefore we now assign H'_k instead of H' to each foreground component of the scene. H'_k is computed thanks to a module sc following the equation: $H'_k = H' \times (1 + sc(z_0, \theta_k, z_k))$, sc is further described in table A13. We summarize all hyper-parameters used for the training of the model in table A14. For evaluation we sample z_0 from $\mathcal{U}([-1,1]^{N_b})$, expect for REALTRAFFIC video model where we used $\mathcal{U}([-0.5,0.5]^{N_b})$ where this range was optimal for background fidelity.

A6 Datasets

BALLSINBOWL. This dataset is a replica of the two balls synthetic dataset of [6]. It consists of 2500 training sequences and 500 test sequences of two balls of different fixed colour rolling in bowls of various shapes. We count an epoch as 10,000 iterations over the data. In figure A2 we show some sample data from this dataset.

CLEVR. We used the official CLEVR from [16]. We train on data from train and validation set and evaluate on the test set. Both ours and BlockGAN2D were trained on the subset containing 3 to 6 objects and evaluated on the entire test set.

CLEVR3/CLEVR5/CLEVR5-vbg. We use online code provided by the authors[8] to generate CLEVR3, CLEVR5 and CLEVR5-vbg. As done in [1] we generate 100,000 images keep 90,000 for training and 10,000 for testing.

ShapeStacks We use the official release of the ShapeStacks dataset[9]. We use the `default` partitioning provided with the dataset and merge the training and validation splits for a total of 264,384 training images. All FID comparisons are made against 10,000 images randomly sampled from the test set which contains 46,560 images in total. Since the original resolution of the images is 224×224 pixels, we re-scale them to 128×128 before feeding them to our network.

REALTRAFFIC. We recorded 5 hours from Youtube[10] of a live traffic camera at a crossing. The video was then unrolled at 10 fps and manually processed to keep only sequences with a number of

[8] `https://github.com/TitasAnciukevicius/clevr-dataset-gen.`

[9] `https://shapestacks.robots.ox.ac.uk/#data`

[10] `https://www.youtube.com/watch?v=5_XSYlAfJZM`

Table A7: Network architecture for module g and g_v. * indicates modification of g_v

Layer name	Layer Type	Input size	Output size	Activation
FCg_1	Linear	$32 + N_f + 2 + 2^* + N_b$	32	LeakyReLU
FCg_2	Linear	32	32	LeakyReLU
FCg_3	Linear	32	2	Tanh

Table A8: Network architecture for module e_v.

Layer name	Layer Type	Input size	Output size	Activation
FCe_v_1	Linear	$N_f + 2 + N_b$	128	LeakyReLU
FCe_v_2	Linear	128	128	LeakyReLU
FCe_v_3	Linear	128	3×2	Tanh

cars in [1,5]. We kept 560 videos for the training set and 123 in test (80/20 ratio). This dataset will be publicly released.

A7 Qualitative results

We provide additional qualitative generation results. Figure A3 shows a failure case of BlockGAN2D mentioned in the paper. In fact, when the scene is more structured BlockGAN2D fails to be object centric and let the background render the entire scene. In addition figures A4, A6, A7, A8 and A9 provide more samples on every dataset for all the models we trained. In particular we can see that when inter-objects relations are weak in CLEVR5 or CLEVR5-vbg, BlockGAN2D performs qualitatively similar to ours (see figures A6 and A7). However when the scene is more crowded and the objects have higher correlation BlockGAN2D quality decreseases significantly (see figures A4, A8 and A9).

Table A9: Network architecture for the discriminators. Note that the Instance Normalization weights were also subjected to spectral normalization. P and D shares weights until Flatten layer.

Layer name	Layer Type	Input size	Output size	Kernel Size	Stride	Activation	Norm.
Convd_1	Conv	$128 \times 128 \times 3$	$64 \times 64 \times 64$	5×5	2	LeakyReLU	-
Convd_2	Conv	$64 \times 64 \times 64$	$32 \times 32 \times 128$	5×5	2	LeakyReLU	IN/Spec. Norm.
Convd_3	Conv	$32 \times 32 \times 128$	$16 \times 16 \times 256$	5×5	2	LeakyReLU	IN/Spec. Norm.
Convd_4	Conv	$16 \times 16 \times 256$	$8 \times 8 \times 512$	5×5	2	LeakyReLU	IN/Spec. Norm.
Convd_5	Conv	$8 \times 8 \times 512$	$4 \times 4 \times 1024$	5×5	2	LeakyReLU	IN/Spec. Norm.
Flatten	Id	$4 \times 4 \times 1024$	$1 \times 1 \times 16384$	-	-	-	-
$Disc$	Linear	$1 \times 1 \times 16384$	1	-	-	Sigmoid	None/Spec. Norm.
P_{end}	Linear	$1 \times 1 \times 16384$	2	-	-	Tanh	None/Spec. Norm.

Table A12: Comparison of RELATE to its scale augmented version (see appendix A5). FID score (lower is better) for various datasets. X indicates failure of convergence. Overall predicting scale improves FID score except in the REALTRAFFIC dataset where there is a marginal decrease of performance. For CLEVR-3, which features wider of objects sizes, our augmented model is the only one to reach convergence.

RELATE variant	CLEVR-3 General	CLEVR-5 General	CLEVR-5vbg General	CLEVR General	ShapeStacks Ordered	REALTRAFFIC General
Ours	X	70.5	**54.3**	79.7	102.3	**42.0**
Ours + scale	**49.3**	**56.7**	**54.3**	**58.3**	**91.8**	46.8

Table A13: Network architecture for module sc.

Layer name	Layer Type	Input size	Output size	Activation
FCsc_1	Linear	$N_f + 2 + N_b$	32	LeakyReLU
FCsc_2	Linear	32	32	LeakyReLU
FCsc_3	Linear	32	1	Tanh

Table A14: Hyper-parameters for each datasets for the scale augmented model. Epoch nums are the number of epochs we trained for. Model is described in appendix A5.

Dataset	Learning rate	Epoch nums	M	$K_{min} - K_{max}$	H'	N_b	N_f	H''/H sampling range
CLEVR3	0.0001	40	2	2-3	6	1	20	$[-0.6, 0.6]^2$
CLEVR5	0.0001	40	2	2-5	4	1	20	$[-0.6, 0.6]^2$
CLEVR5-vbg	0.0001	40	2	2-5	4	1	20	$[-0.6, 0.6]^2$
CLEVR	0.0001	40	2	3-6	6	1	20	$[-0.6, 0.6]^2$
ShapeStacks	0.001	30	2	2-5	4	5	20	$[-0.6, 0.6] \times [0, 0.6]$
REALTRAFFIC	0.0001	20	2	1-5	6	1	20	$[-0.6, 0.6]^2$

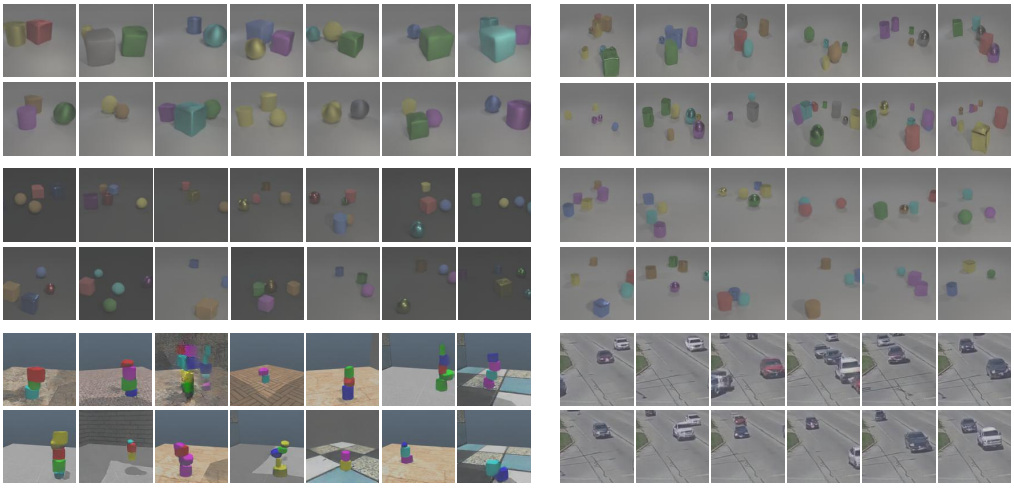


Figure A1: Samples from the scale augmented RELATE. From left to right and top to bottom we display CLEVR-3, CLEVR, CLEVR-5vbg, CLEVR-5, ShapeStacks and REALTRAFFIC. The model makes high fidelity prediction in the case of CLEVR-3 while the objects' scale varies a lot between different images. Similarly predictions on CLEVR comprise more variety of object sizes compared to our main model (see figure A8). Finally on REALTRAFFIC we can see that cars can be rendered at arbitrary points in the road.

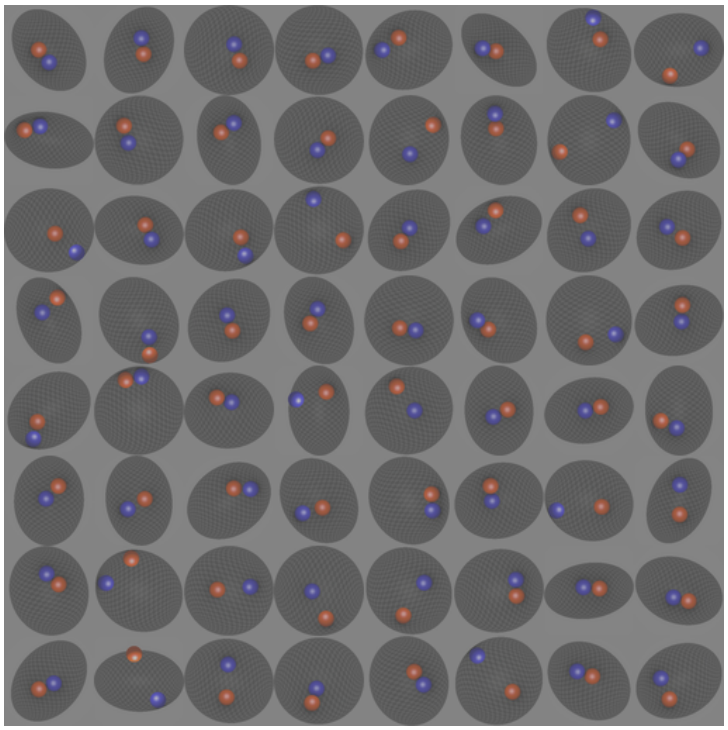


Figure A2: Sample data from BALLSINBOWL. The dataset consists of two balls of different colors rolling in elliptical bowls of various shapes.

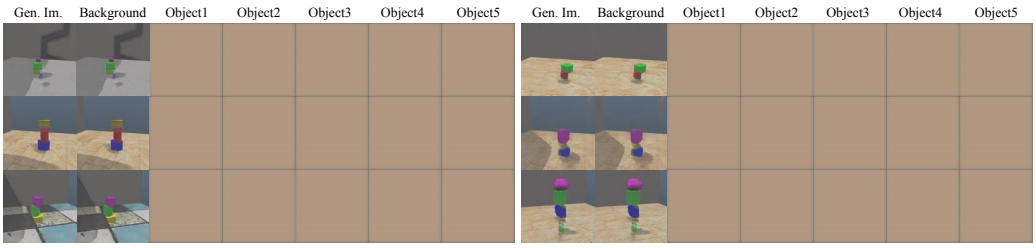
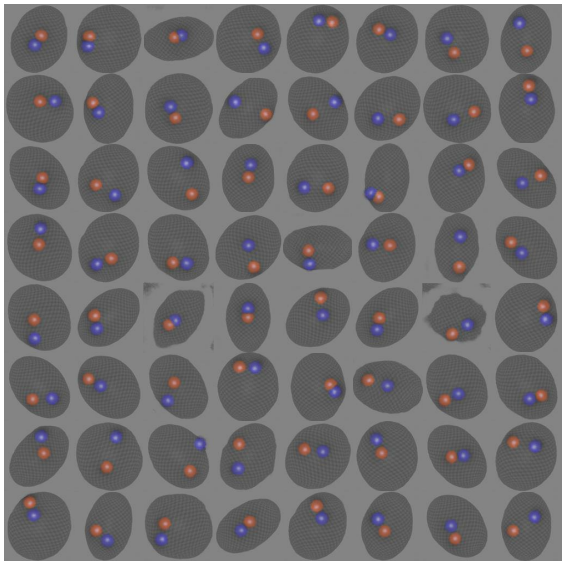
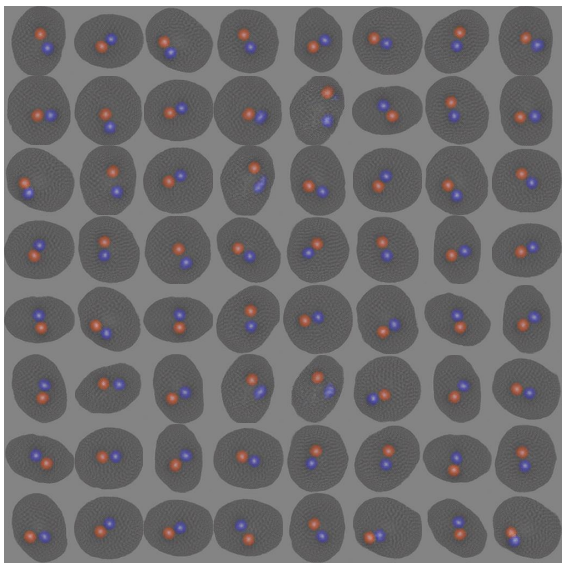


Figure A3: BlockGAN2D scene decomposition on ShapeStacks. We display in order (generated, background, object_1, ..., object_5) for BlockGAN2D model. We see that in this case BlockGAN does not decompose the scene into objects and renders everything in the background instead. This shows how, for structured scenes, prior work fails to capture correlations between objects.

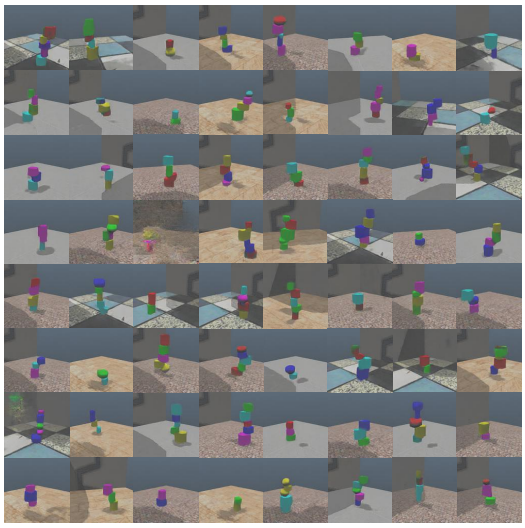


RELATE

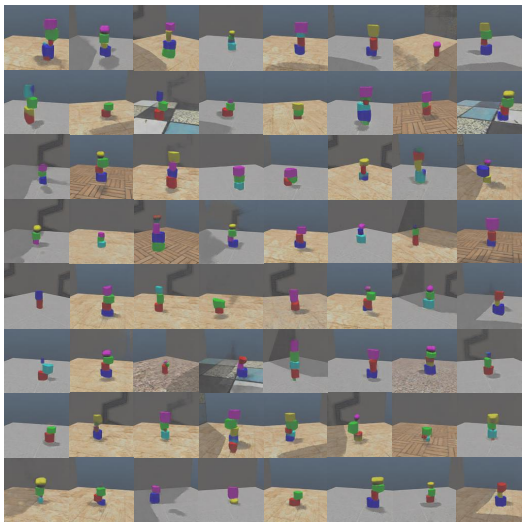


BLOCKGAN2D

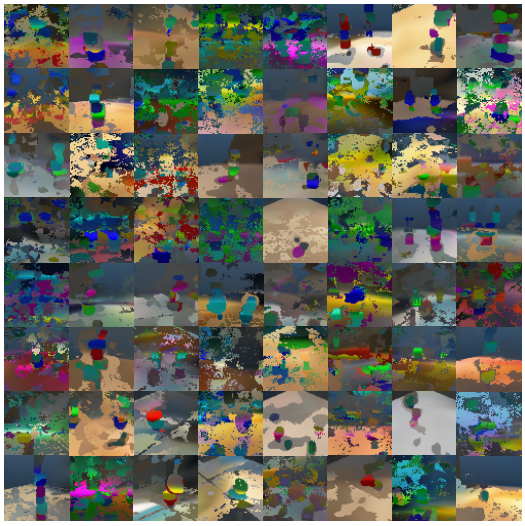
Figure A4: Generated scenes for models trained on BALLSINBOWL. Qualitatively RELATE generates images of higher quality compared to BlockGAN2D[29].



RELATE

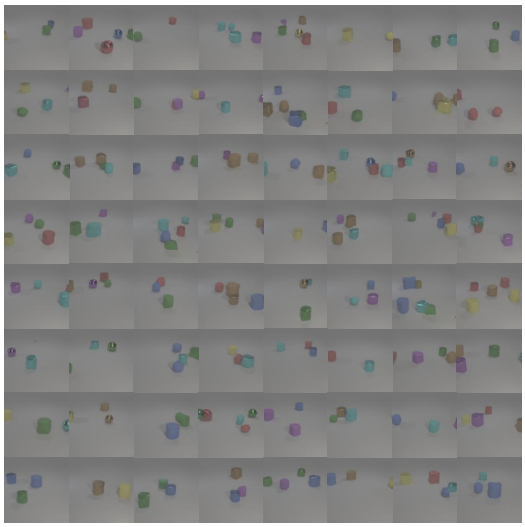


BLOCKGAN2D

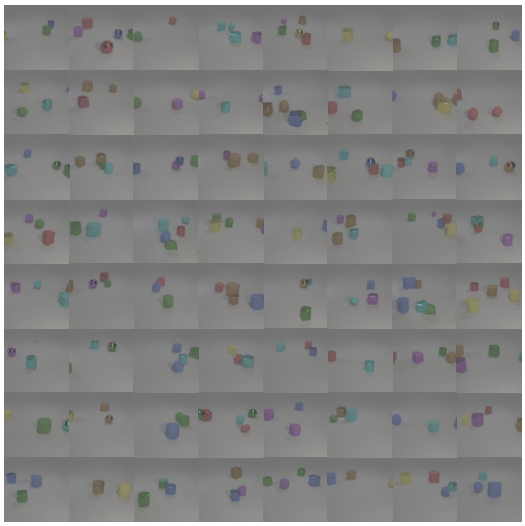


GENESIS

Figure A5: Generated scenes for models trained on ShapeStacks. Despite qualitative similar rendering, BlockGAN2D does not render a scene component-wise as opposed to ours (see figure A3).



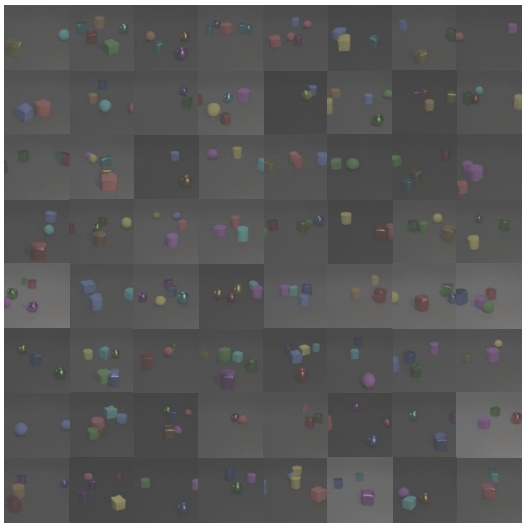
RELATE



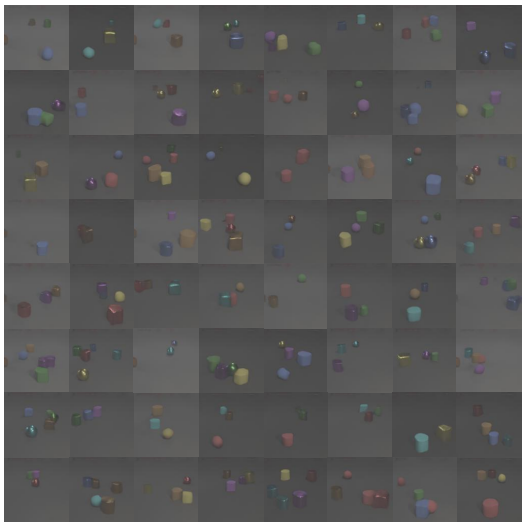
BLOCKGAN2D

GENESIS

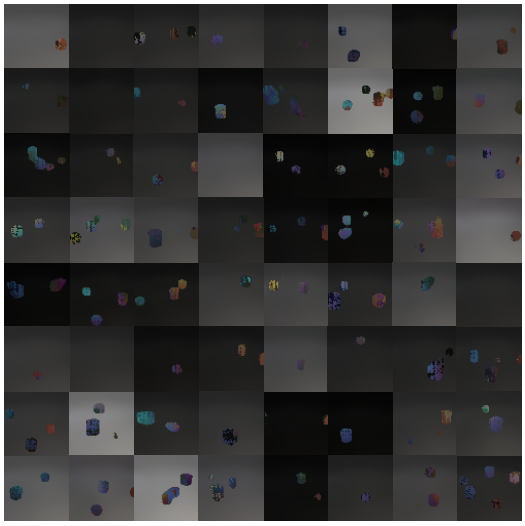
Figure A6: Generated scenes for models trained on CLEVR5. For less crowded scenes our model and BlockGAN2D reach similar performances.



RELATE



BLOCKGAN2D



GENESIS

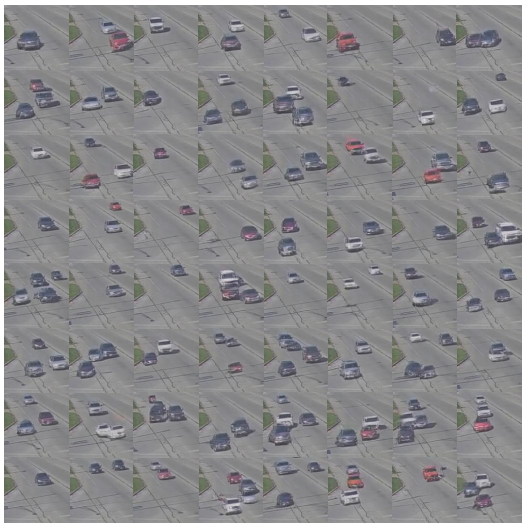
Figure A7: Generated scenes for models trained on CLEVR5-vbg. This scenario reaches similar conclusion as figure A6.

RELATE

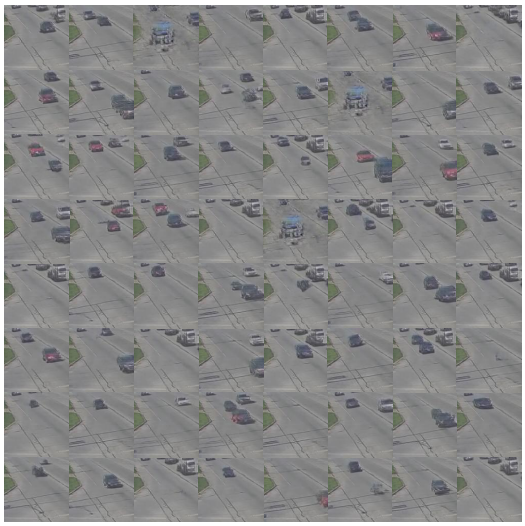
BLOCKGAN2D

GENESIS

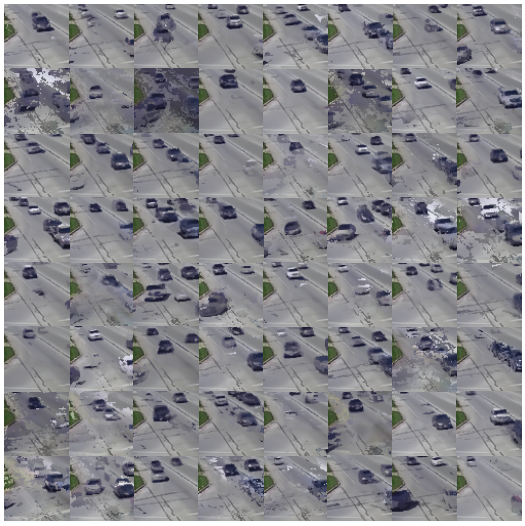
Figure A8: Generated scenes for models trained on CLEVR. When the scene gets more crowded RELATE gets an advantage as it can push objects apart resulting in higher qualitative rendering.



RELATE



BLOCKGAN2D



GENESIS

Figure A9: Generated scenes for models trained on REALTRAFFIC. Our model qualitatively renders higher fidelity images. BlockGAN2D sometimes suffers from background mode collapse (see first, second and fourth rows of second block).

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	RELATE: Physically Plausible Multi-Object Scene Synthesis Using Structured Latent Spaces
Publication Status	□Published □Accepted for Publication x Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style
Publication Details	Sebastien Ehrhardt*, Oliver Groth*, Aron Monszpart, Martin Englecke, Ingmar Posner, Niloy Mitra and Andrea Vedaldi **Submitted** to Advances in Neural Information Processing Systems (2020) *denotes equal contibutions

Student Confirmation

Student Name:	Sebastien Ehrhardt
Contribution to the Paper	• Development of the idea. • Development of the code library and structure. • Ablation study. • Hyper-parameter search and evaluation of the main method on all dataset. • Data collection of RealTraffic. • Writing paper. • Writing supplementary material.
Signature 	Date 27.07.2020.

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof. Andrea Vedaldi	
Supervisor comments	
Signature Andrea Vedaldi	Date 31/07/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 7

Semi-supervised learning with scarce annotations

As we complete our study of intuitve physics we introduce a new part of this thesis dedicated to another concept easily grasped by humans and related to natural intelligence *i.e object recognition*. As mentioned in the introduction, the task of object recognition will be limited to the one of image classification that is, assigning a single class to every image in a set.

In this chapter, we study a first step towards fewer annotations in object recognition: semi-supervision. We focus on the limitation of current semi-supervised classification algorithms in the limit of scarce annotations. In this work, we propose an alternative optimisation scheme based on pseudo-labels that alongside self-supervision performs at par with current *state-of-the-art* algorithms. In particular, we show that this algorithm works increasingly better with better initialisation and reaches *state-of-the-art* performances when combined with other methods. Finally, we also show that self-supervision does not only benefit our algorithm, it also improves performances of all other methods.

This work was presented at the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2020.

Semi-Supervised Learning with Scarce Annotations

Sylvestre-Alvise Rebuffi* Sebastien Ehrhardt* Kai Han*
Andrea Vedaldi Andrew Zisserman
Visual Geometry Group, University of Oxford
{srebuffi,hyenal,khan,vedaldi,az}@robots.ox.ac.uk

Abstract

While semi-supervised learning (SSL) algorithms provide an efficient way to make use of both labelled and unlabelled data, they generally struggle when the number of annotated samples is very small. In this work, we consider the problem of SSL multi-class classification with very few labelled instances. We introduce two key ideas. The first is a simple but effective one: we leverage the power of transfer learning among different tasks and self-supervision to initialize a good representation of the data without making use of any label. The second idea is a new algorithm for SSL that can exploit well such a pre-trained representation. The algorithm works by alternating two phases, one fitting the labelled points and one fitting the unlabelled ones, with carefully-controlled information flow between them. The benefits are greatly reducing overfitting of the labelled data and avoiding issue with balancing labelled and unlabelled losses during training. We show empirically that this method can successfully train competitive models with as few as 10 labelled data points per class. More in general, we show that the idea of bootstrapping features using self-supervised learning always improves SSL on standard benchmarks. We show that our algorithm works increasingly well compared to other methods when refining from other tasks or datasets.

1. Introduction

The success of Deep Learning (DL) in computer vision comes at the cost of collecting large quantities of labelled data. In many applications, data collection is increasingly inexpensive, but data annotation still involves manual and thus expensive labor. Semi-supervised learning (SSL) can significantly reduce the cost of learning new models by using large datasets of which only a small proportion comes with manual labels.

When only some data samples are annotated, one can exploit the structure in the data to infer, implicitly or explicitly, the missing annotations. For example, the class of an image generally does not change if the camera is slightly shifted while maintaining the focus on the same point. The resulting viewpoint change can be approximated by a deformation or warp of the image. A change in illumination, which also generally leaves the image class unchanged, can instead be approximated by a linear transformation of the image range. Hence, an effective way of propagating labels is to enforce consistency in prediction: if one of these perturbations is applied to the image, then the prediction should remain the same.

A simple way to incorporate such invariance while training a deep neural network is to consider image augmentations such as rotations and flips and other perturbations such as Dropout [34], which is injected directly at the level of the features. Such perturbations are used as a form of regularization for supervised learning to avoid overfitting. The idea is that a perturbed image should maintain the same ground-truth label as the original image. In the SSL setting, where most labels are unknown, augmentation can be used to encourage the predictions of unlabelled data instances to be 'consistent', i.e. to remain stable across small variations of the input image, regardless of which specific label the data point takes. If data samples are dense enough, stability to local perturbations may be sufficient to propagate the known labels to the rest of the dataset. Similar ideas are used in many approaches to semi-supervised classification, including the recent works of [16, 21, 35]. Still, performance degrades quickly with a diminishing number of labelled instances per class [24].

We can explain this performance drop as follows. Perturbations can bridge some of the gaps between labelled data points, but the effect is ultimately only local. The ability to transport information across points that are farther apart (since there are fewer of them) depends mainly on the smoothness of the data representation. For example, the Euclidean distance is essentially meaningless in image space, so that transferring labels between images that are close as vectors of pixels is very ineffective. On the other hand, if

*Authors contributed equally

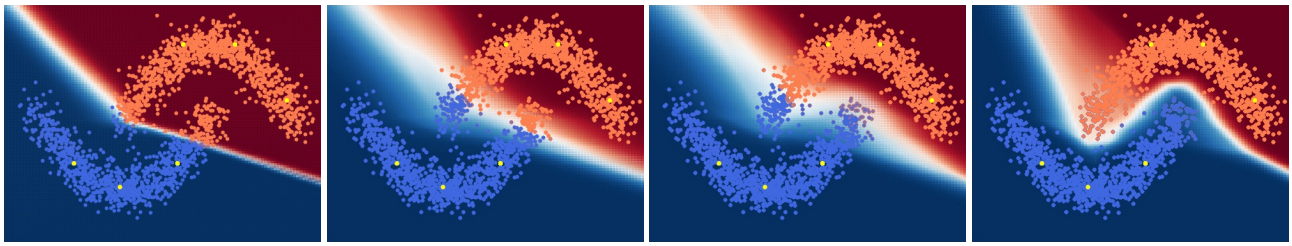


Figure 1. **Consistency loss with the ramp-up strategy.** The Π-model [16] is applied to the 'two moons' problem of [21] and various steps of the optimization are shown. At first, the learned network creates a hard border (first panel) based on the labelled anchor points (in yellow). Then the border is progressively stretched as the consistency loss is enforced more strongly (second and third panels). Overall, the Π-model works well to label connected components of the data space. (Darker regions are more confident; best viewed in color.)

images are first encoded via a representation function in a low dimensional space whose smoothness intrinsically captures some of the relevant invariances, then augmentation can be much more effective.

Unfortunately, simply training a representation such as a deep neural network on a semi-supervised dataset is unlikely to solve the problem. In fact, it is difficult to learn effectively from a label-deficient dataset with a small number of data points with known labels and a very large number of data points with unknown labels. In general, labelled points would work as "anchors": since their label is known with certainty, the neural network is forced to fit those points with confidence, so that the information extracted from them can be reliably propagated. However, when the ratio of labelled vs unlabelled points is very unbalanced, the most likely result is that the neural network would overfit the few labelled points, which would then cease to have an influence on the unlabelled points, thus preventing effective propagation.

We make several contributions to address such shortcomings. The first contribution is a simple but effective one: we propose to use self-supervision to bootstrap a good representation of the data before running our SSL algorithm. Self-supervision defines a pretext-task where a full set of labels can be generated. While these labels are unrelated to the task at hand, we exploit the ability of deep network to *transfer effectively between tasks* and re-use the pre-trained feature to initialize SSL. Note that the data are the same, it is only the task which is being transferred.

The other contribution is a new SSL algorithm that can leverage the bootstrapped representation well. The core idea is that, instead of fitting labelled and unlabelled data points simultaneously, we alternate between two phases of optimization. The information flow between the two phases is carefully controlled to minimize the risk of overfitting. Namely, during *phase one* we fit the labelled data, but only change the final layer of the representation. Then, the resulting classifier is used to generate pseudo-labels for *phase two*, where the unlabelled data are fitted. At this point, the representation is reset to initial state (from the pre-text task) and fine-tuned using the pseudo-labels only. Finally, the fine-tuned representation is passed back to *phase one* to retrain the classifier and obtain new-pseudo labels. In this way, the few available data points are only used to fit a small number of model parameters (a classifier layer only), whereas the large unlabelled dataset is used to fine-tune the representation. We show empirically that this greatly reduces overfitting.

We add other technical contributions to this basic scheme. The most significant in terms of final performance is inspired by cross-validation, Tri-Training [41] and weakly supervised localization [9], and amounts to split our dataset in different subsets, considering only part of the unlabelled data at each cycle of the alternate optimization above.

Empirically, we show that our method achieve close to *state-of-the-art* results in a large number of benchmark cases we test. Furthermore, in addition to showing that our method is effective as a standalone algorithm, we also show two other potential utilities of it. First it is able to refine results obtained from others SSL methods and second it works best among other SSL algorithm when transferring information from one dataset to another. Finally, further studies on architectures and self-supervised task were carried out to assess their importance in improving SSL accuracy.

2. Related work

There exist a vast number of classic works on SSL [4] across many disciplines [7, 31, 36]. In the classification context, one common approach is to optimise the conventional cross-entropy loss on the labelled data together with a regularisation term that propagates information to the unlabelled data. One popular form of 'regularisation' is the consistency of predictions to perturbation [1, 29]. [26] introduced a ladder network that minimises the reconstruction loss between the network outputs from a given sample and its stochastically-perturbed counterpart. [16] simplified the ladder network into two temporal methods: Π-Model and Temporal Ensembling. Both encourage the output of the network for each unlabelled instance to be as similar

as possible between different training epochs, by penalising the inconsistency between current network predictions and past predictions. More recently, the Mean Teacher [35] extended these methods by ensembling over the parameter space: instead of recalling past predictions they run an exponential moving average over the weights of the network to build a Teacher model. The Teacher network is then used to enforce consistency in predictions with the main model, called the Student. Mean Teacher was extended in [19] by a certainty-driven consistency loss that focuses on the unlabelled training samples whose labels are the most uncertain. Similarly, [21] considered instead adversarial perturbations that maximise the change in model prediction. [12] also used this idea together with gradient alignment between labelled and unlabelled data to further improve classification results.

SSL approaches alternative to perturbations include GANs [30, 33]. [6] introduced a memory-assisted deep neural network (MA-DNN) which uses an external memory module to maintain the category prototypes and provide guidance for learning with unlabelled data. Alternatively, [20] uses a label graph to refine the Mean Teacher's model predictions. When using graphs, label propagation [13, 18, 42] is also commonly used. [17] proposed to iteratively assign pseudo-labels for unlabelled data and pick the high-confidence assignments as training data in the subsequent learning steps, combined with the entropy loss to further regularise training. The idea of pseudo labelling is also used alongside with interpolation in [37]. Tri-training [41] and its deep network extension Tri-Net[5] use co-training [3] to generate three classifiers from three different portions of the data and let them label the unlabelled data. [24] shows that consistency is generally the most resilient method as the number of labelled instances decreases, but all such methods suffer significantly when this number is decreased past a certain point.

Transfer learning is not often used in SSL. [8] and [24] used a pre-trained network to fine-tune it on a restricted part of the labelled set. Closer to our approach, [40] also combined both SSL algorithms and transfer learning. However, their approach used a pre-trained network from ImageNet[28] to train SSL methods on non-standard datasets for SSL. Our work extends their study to transfer learning from self-supervision and propose a method that surpasses current SSL technique when transferring representation across dataset.

In addition, we notice very recent concurrent works [2, 32] showing promising results. However, none of them focuses on SSL with scarce annotations.

3. Method

We introduce a SSL algorithm that can work effectively when there are only very few annotated data points, including avoiding supervised pre-training on large labelled datasets. The algorithm starts by a *preparation phase* (section 3.1, line 1 in algorithm 1) that initializes the weights of the model using self-supervision, followed by alternating between two phases (section 3.2, lines 5-9 in algorithm 1), where in *phase one* a subset of the weights are fitted to the few available labels and in *phase two* the whole model is re-trained from scratch using pseudo-labels on all the data.

Algorithm 1 Proposed Alternative Optimisation Algorithm

1: **Preparation phase:**
2: Train a self-supervised method on the whole dataset and freeze the first blocks' weights. Replace the last layer with one dimensioned for the classification task. The trainable weights now form a network N_t
3: **Main Loop:**
4: **for** $i \in \{1, \ldots, N\}$ **do**
5: **Supervised-training**: Fine-tune N_t classification layer on the labelled subset with cross-entropy loss $\mathcal{L}_{labelled}$.
6: **Labels assignement**: Use N_t to assign a label y_i to each unlabelled sample x_i.
7: **Dataset split**: Create a training set T_i from a random split of the unlabelled data.
8: **Restart**: Reassign N_t to the weights extracted from the preparation phase.
9: **Unsupervised-training**: Train N_t on the unlabelled metaset T_i with consistency and pseudo labelling loss: $\mathcal{L}_{unlabelled} = 0.5\mathcal{L}_{temp} + 0.5\mathcal{L}_{pseudo}$.
10: **end for**

3.1. Transfer learning and self-supervision

SSL algorithms risk overfitting the available labelled data points, especially when there is ony a small number of them. Data augmentation and other forms of regularisation can help to some extent, but their effectiveness is limited. A more effective solution is to pre-train the model on a much larger dataset, adding to the information contained in the few labelled data points. Then transfer learning can be used to fine-tune the pre-trained model using the labelled data.

The most common form of pre-training is to use a large, external labelled dataset such as ImageNet. In this work, however, we focus primarily on the case in which labels are scarce, including in the pre-training phase. Furthermore, we wish to avoid the use of external data altogether.

In order to do so, instead of using an external labelled dataset for pre-training, we propose to use instead *self-supervision*. Self-supervision uses a pretext task defined on the available data, both labelled and unlabelled, to bootstrap the model. The advantage is that self-supervision does not require any label nor external data.

Empirically, we show that self-supervised pre-

initialization is very good. In particular, excellent performance can be obtained by freezing most of the model parameters to their pre-initialized values and use our SSL algorithm to fine-tune only a small subset of them.

3.2. Alternate optimisation

Given a pre-initialized model, most SSL algorithms fine-tune it using mini-batches containing a mix of labelled and unlabelled data. They use a sum of two losses, the first one enforcing the correct classification of the labelled samples and the second one enforcing a form of prediction consistency for the unlabelled samples. This consistency term usually captures the fact that neighbor data points are likely to have the same label. Often, a ramp-up is used for the consistency loss [16, 35], so that in the first iterations of training the model focuses on fitting the labelled data.

The effect of this ramp-up can be visualized using the 'two moons' of [21], a simple 2D toy clustering problem. In fig. 1, we apply the Π-model algorithm of [16] to this dataset. We see that, before the consistency loss is applied, the network learns a simple boundary that fits the labelled anchor points with tightly. Then, when the consistency loss is applied, the network stretches the boundary to satisfy the consistency criterion. This causes the classifier to become less certain near the decision boundary, but more accurate overall.

A drawback of this approach is that, since both losses are optimized jointly, they must be carefully balanced by the choice of appropriate hyper-parameters. This is particularly true when there are very few label samples, which causes a large imbalance between the two loss terms. Furthermore, since mini batches are generally evenly split between labelled and unlabelled samples, the few labelled samples will be seen very frequently by the optimizer, which will thus overfit them.

Inspired by incremental learning techniques that try to avoid overfitting on past exemplars while learning new classes [27], we propose to disentangle the two losses by alternating the optimisation of labelled and unlabelled data points while maintaining soft constraints between them. This method reduces the number of hyper-parameters and avoids strong overfitting of the labelled data while still extracting useful knowledge from the labelled set.

In practice, after the preparatory phase of section 3.2 (l.1 in algorithm 1), each task is learnt separately in an alternate fashion with a regularisation term that works as a soft constraint borrowing information from the other task.

***Phase one*: fitting the labelled data.** In the first part of our training, we optimise the cross-entropy loss on the labelled set $\mathcal{L}_{labelled}$ and train the model for a few epochs. In this part we only fine-tune the final classification layer of the network (l.5 in algorithm 1). This way the labelled samples benefit the most from the feature learnt during optimisation on the unlabelled set while not modifying the intermediate representation. We then use this trained network to assign soft or hard pseudo-labels to the unlabelled data. These pseudo-labels are used in the next phase to fit the unlabelled data (l.6 in algorithm 1).

***Phase two*: fitting the unlabelled data.** In the second phase we reset the model with the parameters learnt during the preparatory phase of section 3.2 (see l.8 in algorithm 1). Then, we fine-tune the whole architecture on the unlabelled set using a loss $\mathcal{L}_{unlabelled}$ that is a weighted average of a term $\mathcal{L}_{pseudo}$ fitting the pseudo-labels estimated in the first phase, and a second temporal consistency term $\mathcal{L}_{temp}$, borrowed from the Π-model of [16]. The idea of the second term is to match the probability distribution p_i^{t-1} obtained from the network for a sample x_i at epoch $t-1$ with the probability p_i^t assigned at iteration t to the same sample. We use the KL-divergence between p_i^{t-1} and p_i^t during *phase two*. (l.9 in algorithm 1). The loss $\mathcal{L}_{pseudo}$, besides being a soft constraint tying *phase two* to *phase one*, can also be viewed as another consistency condition [40] that also helps reduce the entropy on the final prediction.

Information flow. Note that the model is reinitialised every time *phase two* is entered. This is done to avoid getting stuck in local minima. Hence, the information that flows from *phase one* to *phase two* is only the pseudo-labels on the unlabelled data, whereas the information that flows from *phase two* back to *phase one* are the weights of the network fine-tuned on the unlabelled data.

While these choices may seem ad-hoc, they constitute the primary reason why our algorithm is able to work well in a low-labelled-data regime by avoiding overfitting. Thanks to this technique, the algorithm focuses on fitting one loss at a time while still allowing information to be exchanged between the two subtasks. We show in section 4 this approach robustly improve the quality of the overall classifier at every step.

3.3. Dataset split

One potential pitfall of alternating optimisation is the drift that may be induced by training for a long time on the unlabelled data. Inspired by cross-validation and co-training [3], we also suggest to split the dataset into several parts. However, differently from [3] we do not train different models from each split. Instead, at the beginning of each training cycle we randomly choose two-thirds of the unlabelled training data to learn from (l.7 in algorithm 1). The rest of the unlabelled data is held out to reduce the risk of overfitting to it in the current training cycle. This way, the held-out samples are more likely to have their labels swapped during pseudo-label generation in the next cy-

cle. Roughly two-thirds of this "fresher" held-out data will be used to fit the model in the next cycle. The benefits of this reshuffling procedure diminish as training nears completion, so all the data is used in the final few phases of training. In section 4 we demonstrate empirically the benefits of this approach.

4. Experiments

Datasets. Following [6] we mostly use standard image classification tasks suitably modified to the semi-supervised context. Each dataset is randomly split in two sets: one small set containing images with their corresponding labels evenly divided among classes and another set with the rest of the image without annotations. For each dataset we evaluated our method on the publicly available test set. We used the same datasets as [6]:

SVHN [23]. A Street View House Numbers dataset including 10 classes (0-9) of coloured digit images from Google Street View. The classification task is to recognise the central digit of each image. We use the format-2 version that provides cropped images sized at 32×32, and the standard 73,257/26,032 training/test data split. We centered and scaled all images RGB channels based on the extracted dataset statistics and augmented the data with random cropping.

CIFAR-10 [15]. A natural images dataset containing 50,000 training and 10,000 test image samples from 10 object classes. Images have a 32×32 resolution and are evenly divided among classes. We again pre-processed the data with centering and scaling and augmented the data with random cropping and flipping.

CIFAR-100 [15]. A dataset (with the same image size as CIFAR-10) containing 50,000/10,000 training/test images from 100 more fine-grained classes. We used the same preprocessing and data augmentation as CIFAR-10.

Baselines. We used Mean-Teacher [35] as the main baseline to experiment alongside with our proposed method. Additionally we also used MA-DNN [6] on a more restricted set of experiments. For both we used publicly available code provided by the authors[1].

Implementation details. Unless stated otherwise, we used the ResNet-18 [11] model with then average pooling and skip connections. We chose RotNet [8] as the self-supervision method as it is easy to implement and integrate in any pipeline. For all networks we trained this proxy task on 200 epochs with a step-wise decaying schedule. A more thorough study is done in section 4.6. All networks are trained using Nesterov accelerated gradient method [22]. We use the same hyper-parameters when training with and without transfer learning, which have every time been tuned on the specific task.

[1] https://github.com/CuriousAI/mean-teacher; https://github.com/yanbeic/semi-memory

4.1. Impact of self-supervision

We first compare Mean-Teacher and our algorithm on standard classification benchmarks. For every dataset we trained the network with 10/25/50/100 labels per class, with an extra experiment of 400 labels per class on CIFAR-10 as is common practice [6, 16, 35]. In every experiment we keep the same split of the data for each method. In the default case, models are trained from scratch. For every method we also used transfer learning from self-supervision, with only the last two blocks and the last classification layer used for semi-supervised training (Legends containing 'with T.L.'). Results are shown in fig. 2.

This experiment clearly demonstrates the importance of transfer learning as the number of labelled data per class decreases. For instance, on CIFAR-10, transfer learning from self-supervision used with semi-supervision makes an improvement of over 50 points in classification error for both our method and Mean Teacher. On SVHN and CIFAR-10 the performance from 100 labels per class to 10 labels per class decreases by less than 5 points in error. Performances on CIFAR-100 dropped by a larger amount when having fewer labels due to the higher complexity of the dataset, although the performance deteriorate noticeably more slowly when using self-supervision pre-training. Overall this means that self-supervision enables networks to train with competitive results in the extremely low data setting where standard SSL methods fail. We also emphasise that we only use the dataset provided and did not use any regularisation method and very limited data augmentation. This is particularly crucial and demonstrates that a carefully chosen self-supervision method can be sufficient to extract very good features which are sufficient to train good models with few labels.

For every dataset we see in fig. 2 that our method and Mean-Teacher obtain similar performance when they both use transfer from self-supervised networks. However our method needs a more careful initialisation as its performance dropped for models trained from scratch. This can easily be understood, as fine-tuning on the labelled set would be more efficient with either an increased number of labelled data or better intermediate representation. In the rest of the paper we call a 'good' representation of an image, a representation from which it is easier to discriminate different classes.

In addition, we show that self-supervision can also improve the state-of-the-art performance of a more recent SSL method. To this end we use MA-DNN [6] with the 10-layers model proposed in their paper. Again we used RotNet [8] as the self-supervision method and only fine-tune the last

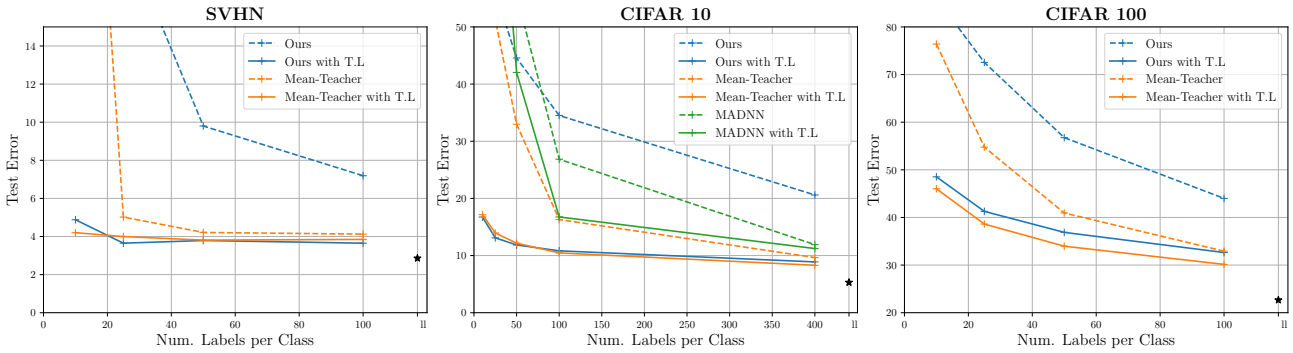


Figure 2. **Impact of Tranfer learning for SSL classification benchmark.** For each dataset we vary the number of labelled data per class in $10, 25, 50, 100$ with an additional experiment on CIFAR-10 with 400 labels per class. We note that for every method, models benefit from self-supervision. The 'with T.L' indicates a model pretrained with RotNet[8]. In addition we can see that our method is getting competitive results with Mean Teacher. The star point '*' denotes our network trained with supervision only on the full training set. The shown results are the average of 10 runs per setting.

block and linear layer on the SSL task. We see in fig. 2 that again, transfer from self-supervision consistently achieves the best performance, improving their *state-of-the-art* results on CIFAR-10 from 11.9% error to 11.1% using 400 labels per class. With the hyper-parameters given in their paper, we demonstrate again that under equivalent settings, self-supervision is beneficial to SSL. Finally we found that MA-DNN was more sensitive to the number of labels per class. This means that building a reliable template of every class requires more annotations, while we observed that consistency based methods are seemingly showing better robustness overall.

4.2. Ablation study

To assess the impact of every component of our method we also conducted an ablation study on the CIFAR-10 SSL task with 10 labels per class.

We first measure the gain of using random splitting of the unlabelled set through training cycles as opposed to using the whole set. We observe in fig. 3 that random dataset splitting allows much faster progress in the course of training. When only using pseudo-labelling on the unlabelled set and no temporal consistency, this technique also leads to a better optimum.

Then we analyse the effect of the consistency in the training cycle. The dashed curves correspond to cases where the temporal consistency is not used. It means that, when training on the unlabeled set, we only optimise the cross-entropy on the pseudo-labels previously assigned during the training phase on the labelled set. Interestingly, we also note that the performance is still increasing over cycles even without an explicit consistency loss. In fact, this corroborates the statement of [40], that the assigned pseudo-labels act as an implicit consistency across samples (instead of the explicit consistency across augmented versions of the same sample) which improves the representation for the next training phase on the labelled set.

Finally we can observe for the different settings that the performance is steadily increasing over cycles. The training indeed does not stall after a few cycles but instead keeps improving the feature representation over the course of alternating optimisation cycles. Alternating optimisation therefore refines the results every cycle with each optimisation helping the next one to reach a better optimum. In addition, we conduct another ablation experiment skipping re-initialization (line 8 in algorithm 1). Compared to the two-phase algorithm with re-initialization, the accuracy drops by 12 points on CIFAR-10 and even 39 points on CIFAR-100. When re-initialization is turned on, $10\times$ more samples switch labels every assignment cycle, thus avoiding local minima. To the best of our knowledge, other existing SSL

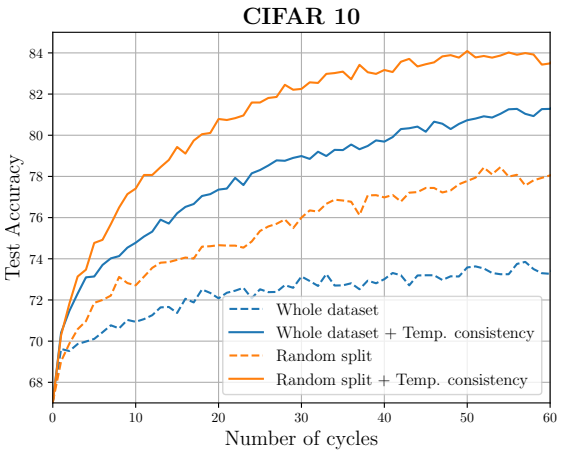


Figure 3. **Ablation study.** Starting from a pre-trained model, our alternating optimisation method makes steady progress over cycles. Dataset random splitting is also effectively speeding-up training. Default case uses $\mathcal{L}_{unlabelled} = \mathcal{L}_{pseudo}$. 'Temp. consistency' indicates that we use both $\mathcal{L}_{temp}$ and $\mathcal{L}_{pseudo}$.

	CIFAR-10	CIFAR-100	
Labels per class	10	10	25
MT	32.4	23.6	45.7
MT + Self-Sup	86.4	53.7	60.7
MT + Self-Sup + Refinement	**87.7**	**58.3**	**63.9**

Table 1. **Our method applied as a refinement of Mean Teacher ('MT').** For different numbers of labels per class our method always improve the testing accuracy by a big margin.

methods do not have any specific mechanism against representation forgetting when doing transfer learning.

4.3. A refining SSL algorithm

In fig. 2 we note that our alternating training method is particularly efficient when using a good representation rather than starting from scratch. Hence we hypothesise that our method could make an even bigger difference when starting from a better network representation. One way of doing it is to build our method on top of others in order to refine their results. To validate this hypothesis, we apply our method as a refinement on CIFAR-10 and CIFAR-100 datasets for networks pre-trained with Mean-Teacher [35]. It should be noted that, as the starting features are already competitive, we do not use the dataset random splitting in our training procedure and we choose soft labels assignment in l.6 of algorithm 1.

In table 1, we see that in every case our method used as refinement consistently improves the test accuracy. The synergy of alternating training is enabling the network to gain as much as 4.6 points in test accuracy on CIFAR-100 and 1.3 points on CIFAR-10. The refinement technique combined with self supervision allows us on CIFAR-10 to almost match previous *state-of-the-art* results with methods trained with 400 labels per class (11.9% reported in [6]) whereas we only used 10 labels. These results confirm our previous hypothesis and motivate us to see the impact of increasingly better representations in the next section.

4.4. Transfer learning from different tasks

So far we have only used the dataset at hand to extract our intermediate representation with self-supervised learning. In this section we use a different type of transfer learning used in [40]: we learn our representation on a different classification dataset. To this end, we use networks pretrained on ImageNet [28]. As is done in the other sections, we then fix the first two blocks of our ResNet-18 network and train the rest of the weights on the hardest semi-supervised learning task of 10 labels per class using either Mean Teacher or our alternating optimisation method (see section 3). We experimented with CIFAR-10 and CIFAR-100 with pretraining on a downsampled 32×32 version of ImageNet, and also MIT Indoor-67 [25], and 'Places10' where we only kept 10 classes from Places [39] using a pretraining on the standard version of ImageNet. Results are shown in table 2 where we perform 10 runs per setting. Fine-tuning on the full set is given as a reference.

	C-10	C-100	P-10	MIT-67
Fully Sup.	96.6	82.5	83.3	75.5
Labelled Set	82.5±1.0	62.8±0.6	66.4±1.7	55.1±0.8
M.T.[35]	88.0±1.5	63.3±0.7	72.8±1.2	58.8±1.3
Ours	**93.5±0.4**	**69.2±0.4**	**75.8±1.7**	**63.0±0.7**

Table 2. **Transfer Learning: Test accuracy on the target task.** Transfer is from training on ImageNet. We compare our method with Mean Teacher (M.T.) and supervision on the labelled set only ('Labelled set') for 10 labelled sample per class. Our method consistently outperforms both method on this task and achieves results nearly as good as the one obtained by training on the full labelled set.

We note that our method consistently outperforms Mean Teacher for such scenarios. This result corroborates our assumption in the previous section that our method performs increasingly well with better representation. In fact, as also noted in [24], since ImageNet has redundant classes with the datasets, we can consider the representation learnt from ImageNet as an upper bound. Thus it is not entirely surprising to see our method obtaining nearly as good results as if it were using the complete set with 93.12% accuracy compared to 96.56% when fine-tuning on the full labelled set on CIFAR-10. As our method aims to disentangle the two losses by using distillation to preserve the effect of the very reduced labelled set, its effect is magnified when using a representation as good as the one learnt on ImageNet. In contrast, techniques like Mean Teacher overfit on a small labelled set, which in turns will lead to the degradation of the representation coming from ImageNet.

4.5. Evaluating representation quality with SSL

In the context of SSL for images classification, practitioners usually use shallow networks. If some works [35, 37] also experimented with deeper and wider architectures, then those were generally applied when the number of labels at hand was large enough (4,000 on CIFAR-10, ∼100,000 on ImageNet). So far in this work, we have shown that it is possible to train a deeper architecture by using transfer-learning from self-supervision. In practice, what matters in our context is the quality of the extracted intermediate representation that we will train our model from.

In this section we experiment with two other architectures and use SSL to measure the quality of their representation. As is done in [14], we compare with one fully-convolutional model and a RevNet[10] model. We chose the fully convolutional 10-layers/3-blocks model from [16] which we named 'TempEns'. We provided our own RevNet implementation named 'RevNet-18' aimed to match the ResNet-18 we used in this work. More specifically, the

N	TempEns[16]	RevNet-18[10]	ResNet-18[11]
10	18.4±1.9	20.0±2.4	**16.4±2.6**
50	16.3±0.9	13.7±0.7	**11.9±0.4**
100	14.8±0.6	12.2±0.5	**10.8±0.4**
400	12.8±0.2	10.5±0.4	**8.9±0.2**
Full	6.7	5.8	5.3

Table 3. **Model variation.** Test error on CIFAR-10 with different number of labels per class 'N'. In all cases networks, were pre-trained with RotNet [8]. First two block are then frozen while the rest of the network is fine-tuned on the classification task. 'Full' indicates that we used the full training set with a fully supervised method. All methods uses the same hyper-parameters.

	Self-supervised training stopped at epoch				FT
	60	120	150	200	
Rotation accuracy	74.1	83.5	85.9	86.1	**86.8**
Labelled set	37.6	40.4	**40.9**	40.7	40.4
SSL (our method)	46.5	50.2	50.4	50.7	**52.3**

Table 4. **Rotation accuracy vs Classification accuracy.** Correlation between rotation accuracy and final SSL classification accuracy on CIFAR-100 with 10 labels per class. 'FT' is fine-tuned model for 500 epochs. 'Labelled' set indicates fine-tuning on label set only.

RevNet-18 has four main blocks and uses the same type of downsampling mechanism as our ResNet-18. We pre-trained all models with RotNet and trained our SSL method on CIFAR-10 with different numbers of annotated instances. For all architectures the first two blocks of the networks are fixed during SSL training, as we found it was giving best results for every architecture. We report results in table 3.

We found that the ResNet-18 has better performance over all methods. TempEns surprisingly remains very competitive for all cases with less weights trained than both methods, with good robustness to extreme cases. While RevNet is almost always able to reach ResNet-18 level of performances. With such sparse data, since RevNet has the most parameters to fine-tune it is thereby the most prone to overfit. Overall, the RevNet results slightly contrasts with the one found by [14], where a RevNet representation surpasses ResNet when training with rotation. However, we note that their methods of evaluation differs significantly from ours. These results emphasise that while K-means neighbors or full supervision can be used to evaluate self-supervised learning [14, 38], SSL could also be used as a way to evaluate the intermediate representation learnt from self-supervised learning algorithms.

4.6. Self-supervision performance vs classification accuracy

In general, different self-supervision tasks would lead to different intermediate representation. However one could also wonder how much solving the proxy task could help to find a representation that would be best suited for the classification task. Intuitively, heavy fine-tuning on the proxy task could lead to a more specialized network which would be harder to adapt to the classification task. To verify this, we studied the rotation accuracy impact on the final SSL performance. Our self-supervised method was trained for 200 epochs with a step-wise decaying learning rate starting from 0.1 at epochs 60, 120, and 160. We saved snapshot of the network before decaying the learning rate and measure the performance reached by our method on the CIFAR-100 classification task with 10 labelled instances per class. In addition, we also trained a RotNet for 500 epochs with a learning rate schedule directly extracted from [6]. We summarise the results in table 4.

Overall we found that fine-tuning on the proxy task was beneficial to the SSL algorithm. Interestingly, we note that the accuracy on the labelled set does not entirely correlate with the final SSL accuracy. While the model with best accuracy on the labelled set obtained 50.4% SSL accuracy, the fine-tuned model obtained an overall SSL accuracy of 52.3%. This also corroborates the results found in section 4.5, better accuracy on the complete or partial set does not necessarily mean a better final SSL accuracy. We believe this result also motivates a new method of self-supervised learning evaluation using SSL.

5. Conclusion and discussion

In this paper, we have pushed the limits of SSL to work with very few annotations. We achieved this goal with several innovations. We first proposed to leverage the power of transfer learning among different tasks and self-supervision to provide a good initial feature representation for SSL. We also proposed a new SSL algorithm that can exploit well such a pre-trained representation, achieving state-of-the-art results on a large number of public benchmarks. We showed that self-supervised learning can not only benefit our proposed SSL algorithm, but also existing state-of-the-art methods. Equipped with the representation learned from self-supervised learning, we found that the performance of state-of-the-art methods can be notably boosted, especially the extreme cases with sparsely annotated data. More generally, we found that SSL could be a new and sensitive downstream task for ranking self-supervised learning methods. Besides working as a standalone method showing superior performance, our method can also serve as an effective refinement method on a network pre-trained either with an existing SSL algorithm or another dataset.

Acknowledgments This work is supported by the EPSRC Programme Grant Seebibyte EP/M013774/1, Mathworks/DTA DFR02620, and ERC IDIU-638009.

References

[1] P. Bachman, O. Alsharif, and D. Precup. Learning with pseudo-ensembles. In *NeurIPS*, 2014. 2

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 20190. 3

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Conference on Computational learning theory*, 1998. 3, 4

[4] O. Chapelle and A. Scholkopf, B.and Zien. *Semi-Supervised Learning*. MIT Press, 2006. 2

[5] D.-D. Chen et al. Tri-net for semi-supervised deep learning. In *IJCAI*, 2018. 3

[6] Y. Chen, X. Zhu, and S. Gong. Semi-supervised deep learning with memory. In *ECCV*, 2018. 3, 5, 7, 8

[7] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NeurIPS*, 2009. 2

[8] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 3, 5, 6, 8

[9] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Multi-fold mil training for weakly supervised object localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2409–2416, 2014. 2

[10] A. N. Gomez et al. The reversible residual network: Backpropagation without storing activations. In *NeurIPS*, 2017. 7, 8

[11] K. He et al. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 8

[12] J. Jackson, J.and Schulman. Semi-supervised learning by label gradient alignment. *ArXiv e-prints*, 2019. 3

[13] K. Kamnitsas et al. Semi-supervised learning via compact latent space clustering. In *ICML*, 2018. 3

[14] X. Kolesnikov, A. andZhai and L Beyer. Revisiting self-supervised visual representation learning. In *ArXiv e-prints*, 2019. 7, 8

[15] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Technical report*, 2009. 5

[16] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *ICLR*, 2017. 1, 2, 4, 5, 7, 8

[17] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013. 3

[18] Q. Li, X-M. Wu, and Z. Guan. Generalized label propagation methods for semi-supervised learning. *ICLR*, 2019. 3

[19] Y. Li, L. Liu, and R. Tan. Certainty-driven consistency loss for semi-supervised learning. In *ArXiv e-prints*, 2019. 3

[20] Y. Luo, J. Zhu, et al. Smooth neighbors on teacher graphs for semi-supervised learning. In *CVPR*, 2018. 3

[21] T. Miyato et al. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE TPAMI*, 2018. 1, 2, 3, 4

[22] Y. Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983. 5

[23] Y Netzer et al. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 5

[24] A. Oliver et al. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, 2018. 1, 3, 7

[25] A. Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, 2009. 7

[26] A. Rasmus et al. Semi-supervised learning with ladder networks. In *NeurIPS*, 2015. 2

[27] S.-A. Rebuffi et al. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 4

[28] O. Russakovsky et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015. 3, 7

[29] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NeurIPS*, 2016. 2

[30] T. Salimans et al. Improved techniques for training gans. In *NeurIPS*, 2016. 3

[31] M. Shi and B. Zhang. Semi-supervised learning improves gene expression-based prediction of cancer recurrence. *Bioinformatics*, 27(21):3017–3023, 2011. 2

[32] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *ArXiv e-prints*, 2020. 3

[33] J. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *ArXiv e-prints*, 2015. 3

[34] N. Srivastava et al. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research (JMLR)*, 2014. 1

[35] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017. 1, 3, 4, 5, 7

[36] A. Teichman and S.n Thrun. Tracking-based semi-supervised learning. *The International Journal of Robotics Research*, 31(7):804–818, 2012. 2

[37] Verma.V et al. Interpolation consistency training for semi-supervised learning. In *ArXiv e-prints*, 2019. 3, 7

[38] L. Zhang et al. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *CVPR*, 2019. 8

[39] B. Zhou et al. Learning deep features for scene recognition using places database. In *NeurIPS*, 2014. 7

[40] H. Zhou et al. When semi-supervised learning meets transfer learning: Training strategies, models and datasets. In *ArXiv e-prints*, 2018. 3, 4, 6, 7

[41] Z.-H. Zhou and M. Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge & Data Engineering*, 2005. 2, 3

[42] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002. 3

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Semi supervised learning with scarce annotations
Publication Status	□Published x Accepted for Publication □Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style
Publication Details	Sylvestre-Alvise Rebuffi*, Sebastien Ehrhardt*, Kai Han*, Andrea Vedaldi and Andrew Zisserman Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition DeepVision Workshops (2020) *denotes equal contibutions

Student Confirmation

Student Name:	Sebastien Ehrhardt		
Contribution to the Paper	• Contributed to the development of the ideas. • Related work investigation. • Main experiments on CIFAR-10, CIFAR-100 and SVHN with comparison to related work. • Evaluation of the method's impact on different architecture. • Evaluation with different self-supervised learning performances. • Writing paper. • Answer to reviews (rebuttal).		
Signature		Date	27.07.2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof. Andrea Vedaldi		
Supervisor comments		
Signature 	Date	31/07/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 8

Automatically discovering and learning new visual categories with ranking statistics

If semi-supervised learning is one of the most common tasks to deal with partially annotated data, it is not the only one. In fact, an emerging trend of work (Hsu et al., 2018, 2019; Han et al., 2019) has considered the case of *novel class discovery*. In such a scenario the knowledge acquired on labelled data is transferred to a clustering task. This slightly differs from our previous chapter in semi-supervised learning as we now assume that the labelled data and unlabelled data have no common category. Hence, we are trying to *discover* new categories in the unlabelled set.

In this chapter we introduce a general pipeline to perform novel category discovery. At the core of our system is a novel way to compare features extracted from a CNN. In fact, we show that using ranking statistics of CNN embeddings can be a way to describe the semantics of images. Combined with a cross entropy loss and a consistency loss inspired from the previous chapter our method obtains *state-of-the-art* results on current benchmarks for novel class discovery. In addition, we found that it is possible to learn a global classifier that can predict the category of any data coming from the distribution of the joint set of labelled and unlabelled data.

This work was presented at the International Conference on Learning Representations (ICLR), 2020.

AUTOMATICALLY DISCOVERING AND LEARNING NEW VISUAL CATEGORIES WITH RANKING STATISTICS

Kai Han* **Sylvestre-Alvise Rebuffi*** **Sebastien Ehrhardt***
Andrea Vedaldi **Andrew Zisserman**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{khan,srebuffi,hyenal,vedaldi,az}@robots.ox.ac.uk

ABSTRACT

We tackle the problem of discovering novel classes in an image collection given labelled examples of other classes. This setting is similar to semi-supervised learning, but significantly harder because there are no labelled examples for the new classes. The challenge, then, is to leverage the information contained in the labelled images in order to learn a general-purpose clustering model and use the latter to identify the new classes in the unlabelled data. In this work we address this problem by combining three ideas: (1) we suggest that the common approach of bootstrapping an image representation using the labeled data only introduces an unwanted bias, and that this can be avoided by using self-supervised learning to train the representation from scratch on the union of labelled and unlabelled data; (2) we use rank statistics to transfer the model's knowledge of the labelled classes to the problem of clustering the unlabelled images; and, (3) we train the data representation by optimizing a joint objective function on the labelled and unlabelled subsets of the data, improving both the supervised classification of the labelled data, and the clustering of the unlabelled data. We evaluate our approach on standard classification benchmarks and outperform current methods for novel category discovery by a significant margin.

1 INTRODUCTION

Modern machine learning systems can match or surpass human-level performance in tasks such as image classification (Deng et al., 2009), but at the cost of collecting large quantities of annotated training data. Semi-supervised learning (SSL) (Oliver et al., 2018) can alleviate this issue by mixing labelled with unlabelled data, which is usually much cheaper to obtain. However, these methods still require some annotations for each of the classes that one wishes to learn. We argue this is not always possible in real applications. For instance, consider the task of recognizing products in supermarkets. Thousands of new products are introduced in stores every week, and it would be very expensive to annotate them all. However, new products do not differ drastically from the existing ones, so it should be possible to discover them automatically as they arise in the data. Unfortunately, machines are still unable to effectively learn new classes without manual annotations.

In this paper, we thus consider the problem of discovering new visual classes automatically, assuming that a certain number of classes are already known by the model (Hsu et al., 2018; 2019; Han et al., 2019). This knowledge comes in the form of a *labelled dataset* of images for a certain set of classes. Given that this data is labelled, off-the-shelf supervised learning techniques can be used to train a very effective classifier for the known classes, particularly if Convolutional Neural Networks (CNNs) are employed. However, this does not mean that the learned features are useful as a representation of the *new classes*. Furthermore, even if the representation transfers well, one still has the problem of identifying the new classes in an unlabelled dataset, which is a clustering problem.

We tackle these problems by introducing a novel approach that combines three key ideas (section 2 and fig. 1). The first idea is to pre-train the image representation (a CNN) using all available images, both labelled and unlabelled, using a self-supervised learning objective. Crucially, this objective *does not* leverage the known labels, resulting in features that are much less biased towards the labelled

*indicates equal contribution

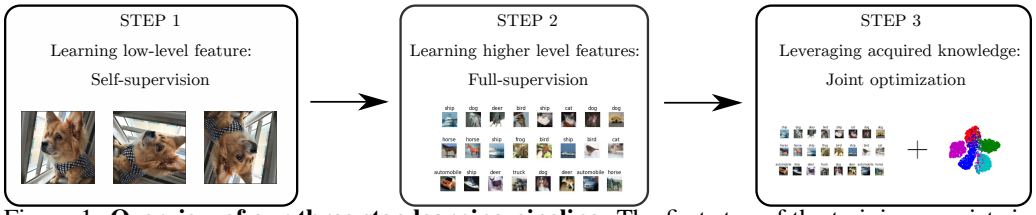


Figure 1: **Overview of our three step learning pipeline**. The first step of the training consists in learning an unbiased image representation via self-supervision using both labelled and unlabelled data, which learns well the early layers of the representation; in the second step, we fine-tune only the last few layers of the model using supervision on the labelled set; finally, the fine-tuned representation is used, via rank statistics, to induce clusters in the unlabelled data, while maintaining a good representation on the labelled set.

classes. Labels are used only after pre-training to learn a classifier specific to the labelled data as well as to fine-tune the deepest layers of the CNN, for which self-supervision is not as effective.

The second idea is a new approach to transfer the information contained in the labelled images to the problem of clustering the unlabelled ones. Information is transferred by sharing the same representation between labelled and unlabelled images, motivated by the fact that the new classes are often similar to the known ones. In more detail, pairs of unlabelled images are compared via their representation vectors. The comparison is done using robust rank statistics, by testing if two images share the same subset of k maximally activated representation components. This test is used to decide if two unlabelled images belong to the same (new) class or not, generating a set of noisy pairwise pseudo-labels. The pseudo-labels are then used to learn a similarity function for the unlabelled images.

The third and final idea is, after bootstrapping the representation, to optimise the model by minimizing a joint objective function, containing terms for both the labelled and unlabelled subsets, using respectively the given labels and the generated pseudo-labels, thus avoiding the forgetting issue that may arise with a sequential approach. A further boost is obtained by incorporating incremental learning of the discovered classes in the classification task, which allows information to flow between the labelled and unlabelled images.

We evaluate our method on several public benchmarks (section 3), outperforming by a large margin all existing techniques (section 4) that can be applied to this problem, demonstrating the effectiveness of our approach. We conclude the paper by summarizing our findings (section 5). Our code can be found at `http://www.robots.ox.ac.uk/~vgg/research/auto_novel`.

2 METHOD

Given an *unlabelled* dataset $D^u = \{x_i^u, i = 1, \dots, M\}$ of images $x_i^u \in \mathbb{R}^{3\times H\times W}$, our goal is to automatically cluster the images into a number of classes C^u, which we assume to be known *a priori*. We also assume to have a second *labelled* image dataset $D^l = \{(x_i^l, y_i^l), i = 1, \dots, N\}$ where $y_i^l \in \{1, \dots, C^l\}$ is the class label for image x_i^l. We also assume that the set of C^l labelled classes is disjoint from the set of C^u unlabelled ones. While the statistics of D^l and D^u thus differ, we hypothesize that a general notion of what constitutes a "good class" can be extracted from D^l and that the latter can be used to better cluster D^u.

We approach the problem by learning an image representation $\Phi : x \mapsto \Phi(x) \in \mathbb{R}^d$ in the form of a CNN. The goal of the representation is to help to recognize the known classes and to discover the new ones. In order to learn this representation, we combine three ideas, detailed in the next three sections.

2.1 SELF-SUPERVISED LEARNING

Given that we have a certain number of labelled images D^l at our disposal, the obvious idea is to use these labels to bootstrap the representation Φ by minimizing a standard supervised objective such as the cross-entropy loss. However, experiments show that this causes the representation to overly-specialize for the classes in D^l, providing a poor representation of the new classes in D^u.

Thus we resist the temptation of using the labels right away and use instead a self-supervised learning method to bootstrap the representation Φ. Self-supervised learning has been shown (Kolesnikov et al., 2019; Gidaris et al., 2018) to produce robust low-level features, especially for the first few layers of typical CNNs. It has the benefit that no data annotations are needed, and thus it can be applied to both labelled and unlabelled images during training. In this way, we achieve the key benefit of ensuring that the representation is initialized without being biased towards the labelled data.

In detail, we first pre-train our model Φ with self-supervision on the union of D^l and D^u (ignoring all labels). We use the RotNet (Gidaris et al., 2018) approach[1] due to its simplicity and efficacy, but any self-supervised method could be used instead. We then extend the pre-trained network Φ with a classification head $\eta^l : \mathbb{R}^d \to \mathbb{R}^{C^l}$ implemented as a single linear layer followed by a softmax layer. The function $\eta^l \circ \Phi$ is fine-tuned on the labelled dataset D^l in order to learn a classifier for the C^l known classes, this time using the labels y_i and optimizing the standard cross-entropy (CE) loss:

$$L_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N} \log \eta^l_{y_i}(z^l_i) \tag{1}$$

where $z^l_i = \Phi(x^l_i) \in \mathbb{R}^d$ is the representation of image x^l_i. Only η^l and the last macro-block of Φ (section 3) are updated in order to avoid overfitting the representation to the labelled data.

2.2 Transfer learning via rank statistics

Once the representation Φ and the classifier η^l have been trained, we are ready to look for the new classes in D^u. Since the classes in D^u are unknown, we represent them by defining a relation among pairs of unlabelled images (x^u_i, x^u_j). The idea is that similar images should belong to the same (new) class, which we denote by the symbol $s_{ij} = 1$, while dissimilar ones should not, which we denote by $s_{ij} = 0$. The problem is then to obtain the labels s_{ij}.

Our assumption is that the new classes will have some degree of visual similarity with the known ones. Hence, the learned representation should be applicable to old and new classes equally well. As a consequence, we expect the descriptors $z^u_i = \Phi(x^u_i)$ and $z^u_j = \Phi(x^u_j)$ of two images x^u_i, x^u_j from the new classes to be close if they are from the same (new) class, and to be distinct otherwise.

Rather than comparing vectors z^u_i, z^u_j directly (e.g., by a scalar product), however, we use a more robust rank statistics. Specifically, we rank the values in vector z^u_i by magnitude. Then, if the rankings obtained for two unlabelled images x^u_i and x^u_j are the same, they are very likely to belong to the same (new) class, so we set $s_{ij} = 1$. Otherwise, we set $s_{ij} = 0$. In practice, it is too strict to require the two rankings to be identical if the dimension of z^u_i is high (otherwise we may end up with $s_{ij} = 0$ for all pairs $(i, j), i \neq j$). Therefore, we relax this requirement by only testing if the *sets* of the top-k ranked dimensions are the same (we use $k = 5$ in our experiments), i.e.:

$$s_{ij} = \mathbb{1}\left\{\text{top}_k(\Phi(x^u_i)) = \text{top}_k(\Phi(x^u_j))\right\}, \tag{2}$$

where $\text{top}_k : \mathbb{R}^d \to \mathcal{P}(\{1, \dots, d\})$ associates to a vector z the subset of indices $\{1, \dots, d\}$ of its top-k elements.

Once the labels s_{ij} have been obtained, we use them as pseudo-labels to train a comparison function for the unlabelled data. In order to do this, we apply a new head $\eta^u : \mathbb{R}^d \to \mathbb{R}^{C^u}$ to the image representation $z^u_i = \Phi(x^u_i)$ to extract a new descriptor vector $\eta^u(z^u_i)$ optimized for the unlabelled data. As in section 2.1, the head is composed of a linear layer followed by a softmax. Then, the inner product $\eta^u(z^u_i)^\top \eta^u(z^u_j)$ is used as a score for whether images x^u_i and x^u_j belong to the same class or not. Note that $\eta^u(z^u_i)$ is a normalized vector due to the softmax layer in η^u. This descriptor is trained by optimizing the *binary cross-entropy* (BCE) loss:

$$L_{\text{BCE}} = -\frac{1}{M^2}\sum_{i=1}^{M}\sum_{j=1}^{M}[s_{ij} \log \eta^u(z^u_i)^\top \eta^u(z^u_j) + (1 - s_{ij}) \log(1 - \eta^u(z^u_i)^\top \eta^u(z^u_j))]. \tag{3}$$

[1] We present to the network Φ randomly-rotated versions Rx of each image and task it with predicting R. The problem is formulated as a 4-way classification of the rotation angle, with angle in $\{0°, 90°, 180°, 270°\}$. The model $\eta \circ \Phi(Rx)$ is terminated by a single linear layer η with 4 outputs each scoring an hypothesis. The parameters of η and Φ are optimized by minimizing the cross-entropy loss on the rotation prediction.

Furthermore, we structure η^u in a particular manner: We set its output dimension to be equal to the number of new classes C^u. In this manner, we can use the index of the maximum element of each vector $\hat{y}_i^u = \text{argmax}_y[\eta^u \circ \Phi(x_i^u)]_y$ as prediction $\hat{y}_i^u$ for the class of image x_i^u (as opposed to assigning labels via a clustering method such as k-means).

2.3 JOINT TRAINING ON LABELLED AND UNLABELLED DATA

We now have two losses that involve the representation Φ: the CE loss L_{CE} for the labelled data D^l and the pairwise BCE loss L_{BCE} for the unlabelled data D^u. They both share the same image embedding Φ. This embedding can be trained sequentially, first on the labelled data, and then on the unlabelled data using the pseudo-labels obtained above. However, in this way the model will very likely forget the knowledge learned from the labelled data, which is known as *catastrophic forgetting* in incremental learning (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Shmelkov et al., 2017; Aljundi et al., 2018).

In contrast, we jointly fine-tune our model using both losses at the same time. Note that most of the model Φ is frozen; we only fine-tune the last macro-block of Φ together with the two heads η^u and η^l. Importantly, as we fine-tune the model, the labels s_{ij} are changing at every epoch as the embedding η^l is updated. This in turn affects the rank statistics used to determine the labels s_{ij} as explained in section 2.2. This leads to a "moving target" phenomenon that can introduce some instability in learning the model. This potential issue is addressed in the next section.

2.4 ENFORCING PREDICTIONS TO BE CONSISTENT

In addition to the CE and BCE losses, we also introduce a consistency regularization term, which is used for both labelled and unlabelled data. In semi-supervised learning (Oliver et al., 2018; Tarvainen & Valpola, 2017; Laine & Aila, 2017), the idea of consistency is that the class predictions on an image x and on a randomly-transformed counterpart tx (for example an image rotation) should be the same. In our case, as will be shown in the experiments, consistency is very important to obtain good performance. One reason is that, as noted above, the pairwise pseudo-labels for the unlabelled data are subject to change on the fly during training. Indeed, for an image x_i^u and a randomly-transformed counterpart tx_i^u, if we do not enforce consistency, we can have $\text{top}_k(\Phi(x_i^u)) \neq \text{top}_k(\Phi(tx_i^u))$. According to eq. (2) defining s_{ij}, it could result in different s_{ij} for x_i^u depending on the data augmentation applied to the image. This variability of the ranking labels for a given pair could then confuse the training of the embedding.

Following the common practice in semi-supervised learning, we use the *Mean Squared Error* (MSE) as the consistency cost. This is given by:

$$L_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(\eta^l(z_i^l) - \eta^l(\hat{z}_i^l))^2 + \frac{1}{M}\sum_{i=1}^{M}(\eta^u(z_i^u) - \eta^u(\hat{z}_i^u))^2, \tag{4}$$

where $\hat{z}$ is the representation of tx.

The overall loss of our model can then be written as

$$L = L_{\text{CE}} + L_{\text{BCE}} + \omega(t)L_{\text{MSE}}, \tag{5}$$

where the coefficient $\omega(t)$ is a ramp-up function. This is widely used in semi-supervised learning (Laine & Aila, 2017; Tarvainen & Valpola, 2017). Following (Laine & Aila, 2017; Tarvainen & Valpola, 2017), we use the sigmoid-shaped function $\omega(t) = \lambda e^{-5(1-\frac{t}{T})^2}$, where t is current time step and T is the ramp-up length and $\lambda \in \mathbb{R}_+$.

2.5 INCREMENTAL LEARNING SCHEME

We also explore a setting analogous to incremental learning. In this approach, after tuning on the labelled set (end of section 2.1), we extend the head η^l to C^u new classes, so that $\eta^l : \mathbb{R}^d \to \mathbb{R}^{C^l+C^u}$. The head parameters for the new classes are initialized randomly. The model is then trained using the same loss eq. (5), but the cross-entropy part of the loss is evaluated on both labelled and unlabelled data D^l and D^u. Since the cross-entropy requires labels, for the unlabelled data we use the *pseudo-labels* $\hat{y}_i^u$, which are generated on-the-fly from the head η^u at each forward pass.

The advantage is that this approach *increments* η^l to discriminate both old and new classes, which is often desirable in applications. It also creates a feedback loop that causes the features z_i^u to be

	CIFAR-10	CIFAR-100	SVHN
Ours w/o Con	82.6±12.0%	61.8±3.6%	61.3±1.9%
Ours w/o CE	84.7±4.4%	58.4±2.7%	59.7±6.6%
Ours w/o BCE	26.2±2.0%	6.6±0.7%	24.5±0.5%
Ours w/o S.S.	89.4±1.4%	67.4±2.0%	72.9±5.0%
Ours full	**90.4±0.5%**	**73.2±2.1%**	**95.0±0.2%**
Ours w/ I.L.	**91.7±0.9%**	**75.2±4.2%**	**95.2±0.3%**

Table 1: **Ablation study.** "w/o Con." means without consistency constraints; "w/o CE" means without the cross entropy loss for training on labeled data. "w/o BCE" means without binary cross entropy loss for training on unlabeled data; "w/o S.S." means without self-supervision.

refined, which in turn generates better pseudo-labels $\hat{y}_i^u$ for D^u from the head η^u. In this manner, further improvements can be obtained by this cycle of positive interactions between the two heads during training.

3 EXPERIMENTS

3.1 DATA AND EXPERIMENTAL DETAILS

We evaluate our models on a variety of standard benchmark datasets: CIFAR-10 (Krizhevsky & Hinton, 2009), CIFAR-100 (Krizhevsky & Hinton, 2009), SVHN (Netzer et al., 2011), OmniGlot (Lake et al., 2015), and ImageNet (Deng et al., 2009). Following Han et al. (2019), we split these to have 5/20/5/654/30 classes respectively in the unlabelled set. In addition, for OmniGlot and ImageNet we use 20 and 3 different splits respectively, as in Han et al. (2019), and report average clustering accuracy. More details on the splits can be found in appendix A.

Evaluation metrics. We adopt *clustering accuracy* (ACC) to evaluate the clustering performance of our approach. The ACC is defined as follow:

$$\max_{g \in \mathrm{Sym}(L)} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left\{\overline{y}_i = g\left(y_i\right)\right\}, \tag{6}$$

where $\overline{y}_i$ and y_i denote the ground-truth label and clustering assignment for each data point $x_i^u \in D^u$ respectively, and $\mathrm{Sym}(L)$ is the group permutations of L elements (this discounts the fact that the cluster indices may not be in the same order as the ground-truth labels). Permutations are optimized using the Hungarian algorithm (Kuhn, 1955).

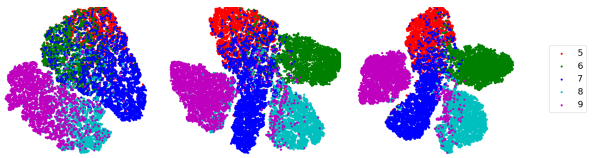
Implementation details. We use the ResNet-18 (He et al., 2016) architecture, except for OmniGlot for which we use a VGG-like network (Simonyan & Zisserman, 2015) with six layers to make our setting directly comparable to prior work. We use SGD with momentum (Sutskever et al., 2013) as optimizer for all but the OmniGlot dataset, for which we use Adam (Kingma & Ba, 2014). For all experiments we use a batch size of 128 and $k = 5$ which we found worked consistently well across datasets (see appendix D). More details about the hyper-parameters can be found in appendix B.

3.2 ABLATION STUDY

We validate the effectiveness of the components of our method by ablating them and measuring the resulting ACC on the unlabelled data. Note that, since the evaluation is restricted to the unlabelled data, we are solving a clustering problem. The same unlabelled data points are used for both training and testing, except that data augmentation (i.e. image transformations) is not applied when computing the cluster assignments. As can be seen in table 1, all components have a significant effect as removing any of them causes the performance to drop substantially. Among them, the BCE loss is by far the most important one, since removing it results in a dramatic drop of 40–60% absolute ACC points. For example, the full method has ACC 90.4% on CIFAR-10, while removing BCE causes the ACC to drop to 26.2%. This shows that that our rank-based embedding comparison can indeed generate reliable pairwise pseudo labels for the BCE loss. Without consistency, cross entropy, or self-supervision, the performance drops by a more modest but still significant 7.8%, 5.7% and 1.0% absolute ACC points, respectively, for CIFAR-10. It means that the consistency term plays a role as important as the cross-entropy term by preventing the "moving target" phenomenon described in section 2.4. Finally, by incorporating the discovered classes in the classification task, we get a further boost of 1.3%, 2.0% and 0.2% points on CIFAR-10, CIFAR-100 and SVHN respectively.

Table 2: **Novel category discovery results on CIFAR-10, CIFAR-100, and SVHN.** ACC on the unlabelled set. "w/ S.S." means with self-supervised learning.

No		CIFAR-10	CIFAR-100	SVHN
(1)	k-means (MacQueen, 1967)	65.5±0.0 %	56.6±1.6%	42.6%±0.0
(2)	KCL (Hsu et al., 2018)	66.5±3.9%	14.3±1.3%	21.4%±0.6
(3)	MCL (Hsu et al., 2019)	64.2±0.1%	21.3±3.4%	38.6%±10.8
(4)	DTC (Han et al., 2019)	87.5±0.3%	56.7±1.2%	60.9%±1.6
(5)	k-means (MacQueen, 1967) w/ S.S.	72.5±0.0%	56.3±1.7%	46.7±0.0%
(6)	KCL (Hsu et al., 2018) w/ S.S.	72.3±0.2%	42.1±1.8%	65.6±4.9%
(7)	MCL (Hsu et al., 2019) w/ S.S.	70.9±0.1%	21.5±2.3%	53.1±0.3%
(8)	DTC (Han et al., 2019) w/ S.S.	88.7±0.3%	67.3±1.2%	75.7±0.4%
(9)	Ours	**90.4±0.5%**	**73.2±2.1%**	**95.0±0.2%**
(10)	Ours w/ I.L.	**91.7±0.9%**	**75.2±4.2%**	**95.2±0.2%**



(a) init (b) epoch 30 (c) epoch 90

Figure 2: **Evolution of the t-SNE during the training of CIFAR-10.** Performed on unlabelled data (i.e., instances of dog, frog, horse, ship, truck). Colors of data points denote their ground-truth labels.

3.3 Novel category discovery

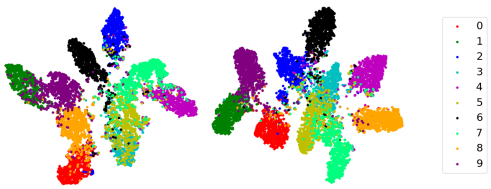
We compare our method to baselines and state-of-the-art methods for new class discovery, starting from CIFAR-10, CIFAR-100, and SVHN in table 2. The first baseline (row 5 in table 2) amounts to applying k-means (MacQueen, 1967) to the features extracted by the fine-tuned model (the second step in section 2.1), for which we use the k-means++ (Arthur & Vassilvitskii, 2007) initialization. The second baseline (row 1 in table 2) is similar, but uses as feature extractor a model trained from scratch using only the labelled images, which corresponds to a standard transfer learning setting. By comparing rows 1, 5 and 9 in table 2, we can see that our method substantially outperforms k-means. Next, we compare with the KCL (Hsu et al., 2018), MCL (Hsu et al., 2019) and DTC (Han et al., 2019) methods. By comparing rows 2–4 to 9, we see that our method outperforms these by a large margin. We also try to improve KCL, MCL and DTC by using the same self-supervised initialization we adopt (section 2.1), which indeed results in an improvement (rows 2–4 vs 6–8). However, their overall performance still lags behind ours by a large margin. For example, our method of section 2.4 achieves 95.0% ACC on SVHN, while "KCL w/ S.S.", "MCL w/ S.S." and "DTC w/ S.S." achieve only 65.6%, 53.1% and 75.7% ACC, respectively. Similar trends hold for CIFAR-10 and CIFAR-100. Finally, the incremental learning scheme of section 2.5 results in further improvements, as can be seen by comparing rows 9 and 10 of table 2.

In fig. 2, we show the evolution of the learned representation on the unlabelled data on CIFAR-10 using t-SNE (van der Maaten & Hinton, 2008). As can be seen, while the clusters overlap in the beginning, they become more and more separated as the training progresses, showing that our model can effectively discover novel visual categories without labels and learn meaningful embeddings for them.

We further compare our method to others on two more challenging datasets, OmniGlot and ImageNet, in table 3. For OmniGlot, results are averaged over the 20 alphabets in the *evaluation* set (see appendix A); for ImageNet, results are averaged over the three 30-class unlabelled sets used in (Hsu et al., 2018; 2019). Since we have a relatively larger number of labelled classes in these two datasets, we follow (Han et al., 2019) and use metric learning on the labelled classes to pre-train the feature extractor, instead of the self-supervised learning. We empirically found that self-supervision does not provide obvious gains for these two datasets. This is reasonable since the data in the labelled sets of these two datasets are rather diverse and abundant, so metric learning can provide good feature initialization as there is less class-specific bias due to the large number of pre-training classes. However, by comparing rows 1 and 5 in table 3, it is clear that metric learning alone is not sufficient for the task of novel category discovery. Our method substantially outperforms the k-means results obtained using the features from metric learning — by 11.9% and 10.6% on OmniGlot and ImageNet respectively. Our method also substantially outperforms the current state-of-the-art, achieving 89.1%

No		OmniGlot	ImageNet
(1)	k-means (MacQueen, 1967)	77.2%	71.9%
(2)	KCL (Hsu et al., 2018)	82.4%	73.8%
(3)	MCL (Hsu et al., 2019)	83.3%	74.4%
(4)	DTC (Han et al., 2019)	89.0%	78.3%
(5)	Ours	**89.1%**	**82.5%**

Table 3: **Novel category discovery results on OmniGlot and ImageNet.** ACC on the unlabelled set.



(a) Ours (b) + incr. learning

Figure 3: **t-SNE on CIFAR-10: impact of incremental Learning.** (a) representation on the labelled and (b) unlabelled CIFAR classes. Colors of data points denote their ground-truth labels. We observe a bigger overlap in (a) between the "old" class 3 and the "new" class 5 when not incorporating Incremental Learning.

and 82.5% ACC on OmniGlot and ImageNet respectively, compared with 89.0% and 78.8% of (Han et al., 2019), thus setting the new state-of-the-art.

3.4 Incremental Learning

Table 4: **Incremental Learning with the novel categories.** "old" refers to the ACC on the labelled classes while "new" refers to the unlabelled classes in the *testing set*. "all" indicates the whole testing set. It should be noted that the predictions are not restricted to their respective subset. Standard deviation can be found in appendix C.

	CIFAR-10			CIFAR-100			SVHN		
Classes	old	new	all	old	new	all	old	new	all
KCL w/ S.S.	79.4%	60.1%	69.8%	23.4%	29.4%	24.6%	90.3%	65.0%	81.0%
MCL w/ S.S.	81.4%	64.8%	73.1%	18.2%	18.0%	18.2%	94.0%	48.6%	77.2%
DTC w/ S.S.	58.7%	78.6%	68.7%	47.6%	49.1%	47.9%	90.5%	72.8%	84.0%
Ours w/ I.L.	**90.6%**	**88.8%**	**89.7%**	**71.2%**	**56.8%**	**68.3%**	**96.3%**	**96.1%**	**96.2%**

Here, we further evaluate our incremental scheme for novel category discovery as described in section 2.5. Methods for novel category discovery such as (Han et al., 2019; Hsu et al., 2019; 2018) focus on obtaining the highest clustering accuracy for the new unlabelled classes, but may forget the existing labelled classes in the process. In practice, forgetting is not desirable as the model should be able to recognize both old and new classes. Thus, we argue that the classification accuracy on the labelled classes should be assessed as well, as for any incremental learning setting. Note however that our setup differs substantially from standard incremental learning (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Shmelkov et al., 2017; Aljundi et al., 2018) where every class is labelled and the focus is on using limited memory. In our case, we can store and access the original data without memory constraints, but the new classes are unlabelled, which is often encountered in applications.

By construction (section 2.5), our method learns the new classes on top of the old ones incrementally, out of the box. In order to compare to methods such as KCL, MCL and DTC that do not have this property, we proceed as follows. First, the method runs as usual to cluster the unlabelled portion of the data, thus obtaining pseudo-labels for it, and learning a feature extractor as a byproduct. Then, the feature extractor is used to compute features for both the labelled and unlabelled training data, and a linear classifier is trained using labels and pseudo-labels, jointly on all the classes, old and new.

We report in table 4 the performance of the resulting joint classifier networks *on the testing set* of each dataset (this is now entirely disjoint from the training set). Our method has similar performances on the old and new classes for CIFAR-10 and SVHN, as might be expected as the split between old and new classes is balanced. In comparison, the feature extractor learned by KCL and MCL works much better for the old classes (e.g., the accuracy discrepancy between old and new classes is 25.3% for KCL on SVHN). Conversely, DTC learns features that work better for the new classes, as shown by the poor performance for the old classes on CIFAR-10. Thus, KCL, MCL and DTC learn representations that are biased to either the old or new classes, resulting overall in suboptimal performance. In contrast, our method works well on both old and new classes; furthermore, it drastically outperforms existing methods on both.

4 RELATED WORK

Our work draws inspiration from semi-supervised learning, transfer learning, clustering, and zero-shot learning. We review below the most relevant contributions.

In semi-supervised learning (SSL) (Chapelle et al., 2006), a partially labelled training dataset is given and the objective is to learn a model that can propagate the labels from the labelled data to unlabelled data. Most SSL methods focus on the classification task where, usually both labelled and unlabelled points belong to the same set of classes. On the contrary, our goal is to handle the case where the unlabelled data classes differ from the labelled data. Oliver et al. (2018) summarizes the state-of-the-art SSL methods. Among them, the consistency-based methods appeared to be the most effective. Rasmus et al. (2015) propose a ladder network which is trained on both labelled and unlabelled data using a reconstruction loss. Laine & Aila (2017) simplifies this ladder network by enforcing prediction consistency between a data point and its augmented counterpart. As an alternative to data augmentation, they also consider a regularization method based on the exponential moving average (EMA) of the predictions. This idea is further improved by Tarvainen & Valpola (2017): instead of using the EMA of predictions, they propose to maintain the EMA of model parameters. The consistency is then measured between the predictions of the current model (student) and the predictions of the EMA model (teacher). More recently (and closer to our work) practitioners have also combined SSL with self-supervision(Rebuffi et al., 2019; Zhai et al., 2019) to leverage dataset with very few annotations.

Transfer learning (Pan & Yang, 2010; Weiss et al., 2016; Tan et al., 2018) is an effective way to reduce the amount of data annotations required to train a model by pre-training the model on an existing dataset. In image classification, for example, it is customary to start from a model pre-trained on the ImageNet (Deng et al., 2009) dataset. In most transfer learning settings, however, both the source data and the target data are fully annotated. In contrast, our goal is to transfer information from a labelled dataset to an unlabelled one.

Many classic (e.g., Aggarwal & Reddy (2013); MacQueen (1967); Comaniciu & Meer (1979); Ng et al. (2001)) and deep learning (e.g., Xie et al. (2016); Chang et al. (2017); Dizaji et al. (2017); Yang et al. (2017; 2016); Hsu et al. (2018; 2019)) clustering methods have been proposed to automatically partition an unlabelled data collection into different classes. However, this task is usually ill-posed as there are multiple, equally valid criteria to partition most datasets. We address this challenge by learning the appropriate criterion by using a labelled dataset, narrowing down what constitutes a proper class. We call this setting "transfer clustering".

To the best of our knowledge, the work most related to ours are (Hsu et al., 2018; 2019; Han et al., 2019). Han et al. (2019) also consider discovering new classes as a transfer clustering problem. They first learn a data embedding by using metric learning on the labelled data, and then fine-tune the embedding and learn the cluster assignments on the unlabelled data. In (Hsu et al., 2018; 2019), the authors introduce KCL and MCL clustering methods. In both, a similarity prediction network (SPN), also used in (Hsu et al., 2016), is first trained on a labelled dataset. Afterwards, the pre-trained SPN is used to provide binary pseudo labels for training the main model on an unlabelled dataset. The overall pipelines of the two methods are similar, but the losses differ: KCL uses a Kullback-Leibler divergence based contrastive loss equivalent to the BCE used in this paper (eq. (3)), and MCL uses the Meta Classification Likelihood loss. Zero-shot learning (ZSL) (Xian et al., 2018; Fu et al., 2018) can also be used to recognize new classes. However, differently from our work, ZSL also requires additional side information (e.g., class attributes) in addition to the raw images.

Finally, other works (Dean et al., 2013; Yagnik et al., 2011) discuss the application of rank statistics to measuring the similarity of vectors; however, to the best of our knowledge, we are the first to apply rank statistics to the task of novel category discovery using deep neural networks.

5 CONCLUSIONS

In this paper, we have looked at the problem of discovering new classes in an image collection, leveraging labels available for other, known classes. We have shown that this task can be addressed very successfully by a few new ideas. First, the use of self-supervised learning for bootstrapping the image representation trades off the representation quality with its generality, and for our problem this leads to a better solution overall. Second, we have shown that rank statistics are an effective method to compare noisy image descriptors, resulting in robust data clustering. Third, we have

shown that jointly optimizing both labelled recognition and unlabelled clustering in an incremental learning setup can reinforce the two tasks while avoiding forgetting. On standard benchmarks, the combination of these ideas results in much better performance than existing methods that solve the same task. Finally, for larger datasets with more classes and diverse data (e.g., ImageNet) we note that self-supervision can be bypassed as the pretraining on labelled data already provides a powerful enough representation. In such cases, we still show that the rank statistics for clustering gives drastic improvement over existing methods.

6 ACKNOWLEDGMENTS

This work is supported by the EPSRC Programme Grant Seebibyte EP/M013774/1, Mathworks/DTA DFR02620, and ERC IDIU-638009.

REFERENCES

Charu C. Aggarwal and Chandan K. Reddy. *Data Clustering: Algorithms and Applications*. CRC Press, 2013.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018.

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *ACM-SIAM symposium on Discrete algorithms*, 2007.

Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *ICCV*, 2017.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006.

Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE TPAMI*, 1979.

Thomas Dean, Mark A. Ruzon, Mark Segal, Jonathon Shlens, Sudheendra Vijayanarasimhan, and Jay Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013.

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

Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *ICCV*, 2017.

Yanwei Fu, Tao Xiang, Yu-Gang Jiang, Xiangyang Xue, Leonid Sigal, and Shaogang Gong. Recent advances in zero-shot recognition: Toward data-efficient understanding of visual content. *IEEE Signal Processing Magazine*, 2018.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *ICLR*, 2018.

Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *ICCV*, 2019.

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

Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Deep image category discovery using a transferred similarity function. *arxiv*, 2016.

Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks. In *ICLR*, 2018.

Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. Multi-class classification without multi-class labels. In *ICLR*, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *CVPR*, 2019.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical report*, 2009.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 2015.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, 2017.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NeurIPS*, 2001.

Avital Oliver, Augustus Odena, Colin Raffel, Ekin D. Cubuk, and Ian J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *NeurIPS*, 2018.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

A. Rasmus, Mathias M. Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *NeurIPS*, 2015.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.

Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Semi-supervised learning with scarce annotations. *arxiv*, 2019.

Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *ICCV*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, 2013.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International Conference on Artificial Neural Networks*, 2018.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 2017.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NeurIPS*, 2016.

Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 2016.

Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. *IEEE TPAMI*, 2018.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016.

Jay Yagnik, Dennis Strelow, David A. Ross, and Ruei sung Lin. The power of comparative reasoning. In *ICCV*, 2011.

Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *ICML*, 2017.

Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, 2016.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. 2019.

A DATASET SPLITS

For CIFAR-10 and SVHN we keep the labels of the five first categories (namely airplane, automobile, bird, cat, deer for CIFAR-10, 0–4 for SVHN) and keep the rest of the data as the unlabelled set. For CIFAR-100 we use the first 80 categories as labelled data while the rest are used for the unlabelled set. Following Hsu et al. (2018; 2019), for OmniGlot, each of the 20 alphabets in *evaluation* set (with 20–47 categories, 659 characters/class) is used as unlabelled data, and all the 30 alphabets in *background* set are used as labelled set (964 characters/class). For ImageNet, we follow Hsu et al. (2018; 2019) to use the 882/118 classes split proposed in Vinyals et al. (2016), and use the three 30-class subsets sampled from the 118 classes as unlabelled sets.

B IMPLEMENTATION DETAILS

In the first self-supervised training step, otherwise mentioned, we trained our model with the pretext task of rotation predictions (i.e., a four-class classification: 0°, 90°, 180°, and 270°) for 200 epochs and a step-wise decaying learning rate starting from 0.1 and divided by 5 at epochs 60, 120, and 160.

In the second step of our framework (i.e., supervised training using labelled data), we fine-tuned our model on the labelled set for 100 epochs and a step-wise decaying learning rate starting from 0.1 and halved every 10 epochs. From this step onward we fix the first three convolutional blocks of the model, and fine-tuned the last convolutional block together with the linear classifier.

Finally, in the last joint training step, we fine-tuned our model for 200/100/90 epochs for {CIFAR-10, CIFAR-100, SVHN}/OmniGlot/ImageNet, which was randomly sampled from the merged set of both labelled and unlabelled data. The initial learning rate was set to 0.1 for all datasets, and was decayed with a factor of 10 at the 170th/{30th, 60th} epoch for {CIFAR-10, CIFAR-100, SVHN}/ImageNet. The learning rate of 0.01 was kept fixed for OmniGlot. For the consistency regularization term, we used the ramp-up function as described in section 2.4 with $\lambda = \{5.0, 50.0, 50.0, 100.0, 10.0\}$, and $T = \{50, 150, 80, 1, 50\}$ for CIFAR-10, CIFAR-100, SVHN, OmniGlot, and ImageNet respectively.

In the incremental learning setting, all previous hyper parameters remain the same for our method. We only add a ramp-up on the cross entropy loss on unlabelled data. The ramp-up length is the same as the one used for eq. (4) and we use for all experiments a coefficient of 0.05. For all other methods we trained the classifier for 150 epochs with SGD with momentum and learning rate of 0.1 divided by 10 at epoch 50.

We implemented our method using PyTorch 1.1.0 and ran experiments on NVIDIA Tesla M40 GPUs. Following (Han et al., 2019), our results were averaged over 10 runs for all datasets, except for ImageNet which was averaged over the three 30-class subsets. In general, we found the results were stable. Our code is publicly available at `http://www.robots.ox.ac.uk/~vgg/research/auto_novel`.

C STANDARD DEVIATION OF INCREMENTAL LEARNING EXPERIMENT IN TABLE 4

Table 5: **Incremental Learning with the novel categories.** "old" refers to the standard deviation ACC on the labelled classes while "new" refers to the unlabelled classes in the *testing set*. "all" indicates the whole testing set. It should be noted that the predictions are not restricted to their respective subset.

	CIFAR-10			CIFAR-100			SVHN		
Classes	old	new	all	old	new	all	old	new	all
KCL w/ S.S.	0.6%	0.6%	0.1%	0.3%	0.3%	0.2%	0.3%	0.5%	0.1%
MCL w/ S.S.	0.4%	0.4%	0.1%	0.3%	0.1%	0.2%	0.2%	0.3%	0.1%
DTC w/ S.S.	0.6%	0.2%	0.3%	0.2%	0.2%	0.2%	0.3%	0.2%	0.1%
Ours w/ I.L.	0.2%	0.2%	0.1%	0.1%	0.3%	0.1%	0.1%	0.0%	0.1%

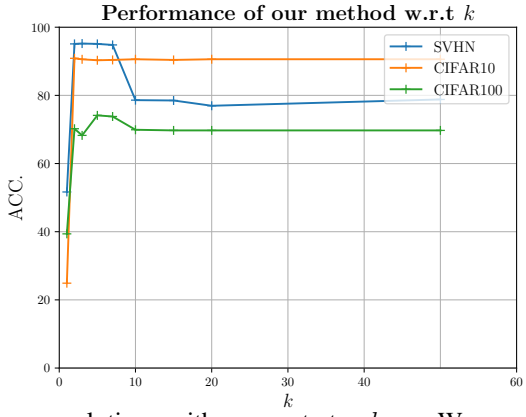


Figure 4: Performance evolution with respect to k. We report results for $k = \{1, 2, 3, 5, 7, 10, 15, 20, 50\}$.

D IMPACT OF k OVER RESULTS

We provide an additional study of the evolution of performances of our method with respect to k. We results on SVHN/CIFAR10/CIFAR100 in fig. 4. We found that $k = \{5, 7\}$ gave the best results overall. We also found that for all values of k except 1 results were in general stable.

E RESULTS WITH AN UNKNOWN NUMBER OF CLASSES

While in our work we assume the number of new classes C^u to be known a priori, this hypothesis can be restrictive in practice. Instead, one can estimate the number of classes in D^u using recent methods such as DTC (Han et al., 2019). In table 6 we compare ACC of KCL (Hsu et al., 2018), KCL (Hsu et al., 2019), DTC(Han et al., 2019) and our method on unlabelled splits of OmniGlot and ImageNet datasets with C^u computed from DTC. We note that our method again reaches the state-of-the-art on ImageNet and is on par with the state-of-the-art on OmniGlot.

Table 6: **Novel category discovery results with unknown** C^u.

No		OmniGlot	ImageNet
(1)	KCL (Hsu et al., 2018)	80.3%	71.4%
(2)	MCL (Hsu et al., 2019)	80.5%	72.9%
(3)	DTC (Han et al., 2019)	**87.0**%	77.6%
(4)	Ours	85.4%	**80.5**%

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Automatically discovering and learning new visual categories with ranking statistics
Publication Status	□Published xAccepted for Publication □Submitted for Publication □Unpublished and unsubmitted work written in a manuscript style
Publication Details	Kai Han*, Sylvestre-Alvise Rebuffi*, Sebastien Ehrhardt*, Andrea Vedaldi and Andrew Zisserman Proceedings of the 8th International Conference on Learning Representations, ICLR (2020) *denotes equal contibutions

Student Confirmation

Student Name:	Sebastien Ehrhardt		
Contribution to the Paper	• Contributed to the main pipeline elaboration (see fig.1 of the paper). • Contributed to the ranking statistics and hard labeling idea. • Ablation of the ranking statistic hyper parameter influence. • Hyper-parameters optimisation. • Evaluation on standard benchmark (SVHN, CIFAR-10, CIFAR-100). • Writing paper. • Answer to reviews.		
Signature		Date	27.07.2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof. Andrea Vedaldi		
Supervisor comments		
Signature 	Date	31/07/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 9

LSD-C: Linearly Separable Deep Clusters

The final chapter of the object recognition part of this thesis examines the lowest level of annotation on data. We are now considering the task of learning without any label *i.e clustering*.

In this chapter we present a novel deep learning method to perform unsupervised clustering. Following our main pipeline initiated in chapter 7 we start from a self-supervised representation. Then, inspired by our previous work in chapter 8 we perform clustering by comparing the nearest neighbours in latent space with distances taken from previous works. Our way of comparing embeddings of neural network for clustering differs from previous works which often proposed to compare the neural networks probability predictions (Chang et al., 2017a). We found that our method combined with a recent data-augmentation method can be used to further improve clustering accuracy and achieve *state-of-the-art* results on unsupervised clustering tasks in four vision benchmarks as well as a text benchmark.

LSD-C: Linearly Separable Deep Clusters

Sylvestre-Alvise Rebuffi* **Sébastien Ehrhardt*** **Kai Han***
Andrea Vedaldi **Andrew Zisserman**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{srebuffi,hyenal,khan,vedaldi,az}@robots.ox.ac.uk

Abstract

We present LSD-C, a novel method to identify clusters in an unlabeled dataset. Our algorithm first establishes pairwise connections in the feature space between the samples of the minibatch based on a similarity metric. Then it regroups in clusters the connected samples and enforces a linear separation between clusters. This is achieved by using the pairwise connections as targets together with a binary cross-entropy loss on the predictions that the associated pairs of samples belong to the same cluster. This way, the feature representation of the network will evolve such that similar samples in this feature space will belong to the same linearly separated cluster. Our method draws inspiration from recent semi-supervised learning practice and proposes to combine our clustering algorithm with self-supervised pretraining and strong data augmentation. We show that our approach significantly outperforms competitors on popular public image benchmarks including CIFAR 10/100, STL 10 and MNIST, as well as the document classification dataset Reuters 10K. Our code is available at https://github.com/srebuffi/lsd-clusters.

1 Introduction

The need for large scale labelled datasets is a major obstacle to the applicability of deep learning to problems where labelled data cannot be easily obtained. Methods such as clustering, which are unsupervised and thus do not require any kind of data annotation, are in principle more easily applicable to new problems. Unfortunately, standard clustering algorithms [8, 11, 39, 42] usually do not operate effectively on raw data and require to design new data embeddings specifically for each new application. Thus, there is a significant interest in automatically learning an optimal embedding while clustering the data, a problem sometimes referred to as simultaneous data clustering and representation learning. Recent works have demonstrated this for challenging data such as images [28, 52] and text [29, 45]. However, most of these methods work with a constrained output space, which usually coincides with the space of discrete labels or classes being estimated, therefore forcing to work at the level of the semantic of the clusters directly.

In this paper, we relax this limitation by introducing a novel clustering method, *Linearly Separable Deep Clustering* (LSD-C). This method operates in the feature space computed by a deep network and builds on three ideas. First, the method extracts mini-batches of input samples and establishes pairwise pseudo labels (connections) for each pair of sample in the mini-batch. Differently from prior art, this is done in the space of features computed by the penultimate layer of the deep network instead of the final output layer, which maps data to discrete labels. From these pairwise labels, the method learns to regroup the connected samples into clusters by using a clustering loss which forces the clusters to be linearly separable. We empirically show in section 4.2 that this relaxation already significantly improves clustering performance.

*indicates equal contribution

Preprint. Under review.

Second, we initialize the model by means of a self-supervised representation learning technique. Prior work has shown that these techniques can produce features with excellent linear separability [4, 17, 23] that are particularly useful as initialization for downstream tasks such as semi-supervised and few-shot learning [15, 43, 55].

Third, we make use of very effective data combination techniques such as RICAP [50] and MixUp [56] to produce composite data samples and corresponding pseudo labels, which are then used at the pairwise comparison stage. In section 4 we show that training with such composite samples and pseudo labels greatly improves the performance of our method, and is in fact the key to good performance in some cases.

We comprehensively evaluate our method on popular image benchmarks including CIFAR 10/100, STL 10 and MNIST, as well as the document classification dataset Reuters 10K. Our method almost always outperforms competitors on all datasets, establishing new state-of-the-art clustering results. The rest of the paper is organized as follows. We first review the most relevant works in section 2. Next, we develop the details of our proposed method in section 3, followed by the experimental results, ablation studies and analysis in section 4. Our code is publicly available at `https://github.com/srebuffi/lsd-clusters`.

2 Related work

Deep clustering. Clustering has been a long-standing problem in the machine learning community, including well-known algorithms such as K-means [39], mean-shift [8], DBSCAN [11] or Gaussian Mixture models [42]. Furthermore it can also be combined with other techniques to achieve very diverse tasks like novel category discovery [22, 12] or semantic instance segmentation [10] among others. With the advances of deep learning, more and more learning-based methods have been introduced in the literature [13, 14, 20, 27, 29, 37, 46, 52, 53]). Among them, DEC [52] is one of the most promising method. It is a two stage method that jointly learns the feature embedding and cluster assignment. The model is first pretrained with an autoencoder using reconstruction loss, after which the model is trained by constructing a sharpened version of the soft cluster assignment as pseudo target. This method inspired a few following works such as IDEC [18] and DCED [19]. JULE [54] is a recurrent deep clustering framework that jointly learns the feature representation with an agglomerative clustering procedure, however it requires tuning a number of hyper-parameters, limiting its practical use. More recently, several methods have been proposed based on mutual information [5, 26, 28]. Among them, IIC [28] achieves the current state-of-the-art results on image clustering by maximizing the mutual information between two transformed counterparts of the same image. Closer to our work is the DAC [3] method, which considers clustering as a binary classification problem. By measuring the cosine similarity between predictions, pairwise pseudo labels are generated from the most confident positive or negative pairs. With the generated pairwise pseudo labels, the model can then be trained by a binary cross-entropy loss. DAC can learn the feature embedding as well as the cluster assignment in an end-to-end manner. Our work significantly differs from DAC as it generates pairwise predictions from a less constrained feature space using similarity techniques not limited to cosine distance.

Self-supervised representation learning. Self-supervised representation learning has recently attracted a lot of attention. Many effective self-supervised learning methods have been proposed in the literature [1, 2, 4, 16, 17, 23]. DeepCluster [2] learns feature representation by classification using the pseudo labels generated from K-means on the learned features in each training epoch. RotNet [17] randomly rotates an image, and learns to predict the applied rotations. Very recently, contrastive learning based methods MoCo [23] and SimCLR [4] have achieved the state-of-the-art self-supervised representation performance, surpassing the representation learnt using ImageNet labels. Self-supervised learning has been also applied in few-shot learning [15], semi-supervised learning [43, 55] and novel category discovery [21], which successfully boosts their performance. In this work we make use of the provably well-conditioned feature space learnt from self-supervised learning method to initialize our network and avoid degenerative cases.

Pairwise pseudo labeling. Pairwise similarity between pairs of sample has been widely used in the literature for dimension reduction or clustering (e.g., t-SNE [38], FINCH [45]). Several methods have shown the effectiveness of using pairwise similarity to provide pseudo labels on-the-fly to train deep convolutional neural networks. In [25], a binary classifier is trained to provide pairwise pseudo labels

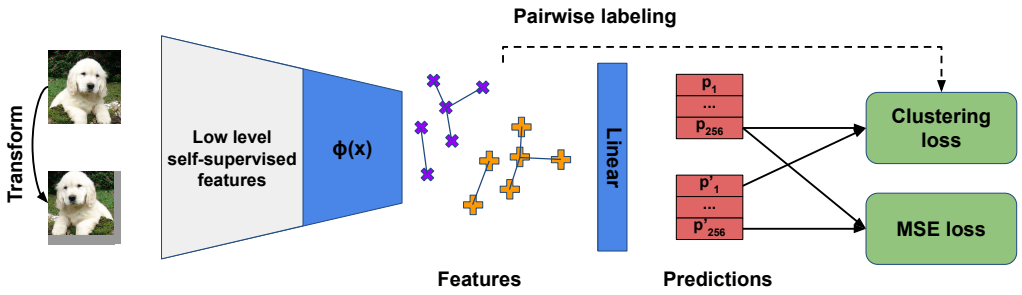


Figure 1: **Overview of LSD-C.** Pairwise labels are extracted at the feature level. They are then used in a clustering loss after the linear classifier. This way, the feature maps will evolve such that connected samples will be grouped in linearly separated clusters. The MSE loss acts a regularizer and enforces the consistency of the cluster predictions when data augmentation is applied.

to train a multi-class classifier. In [21], ranking statistics is used to obtain pairwise pseudo labels on-the-fly for the task of novel category discovery. In [45], the pairwise connection between data points by finding the nearest neighbour is used to cluster images using CNN features. In our method, we compute pairwise labels from a neural network embedding. This way we generate pseudo labels for each pair in each mini-batch and learn cluster assignment without any supervision.

3 Method

Our methods is divided into three stages: (i) self-supervised pre-training, (ii) pairwise connection and clustering, and (iii) data composition. We provide an overview of our pipeline in figure 1. Our method processes each input data batch x in two steps, by extracting features $\mathbf{f} = \Phi(x) \in \mathbb{R}^{N \times D}$ by means of a neural network Φ, followed by estimating posterior class probabilities $\mathbf{p} = \Psi(\mathbf{f}) \in \mathbb{R}^{N \times K}$ by means of a linear layer Ψ and softmax non-linearity. We use the symbol $\mathbf{p}' = \Phi(\Psi(x'))$ to denote the class predictions for the same mini-batch x' with data augmentation (random transformations) applied to it. We use the letters D, K and N to denote the feature space dimension, the number of clusters and the mini-batch size. We now detail each component of LSD-C.

3.1 Self-supervised pretraining

As noted in the introduction, traditional clustering methods require handcrafted or pretrained features. More recently, methods such as [28] have combined deep learning and clustering to learn features and clusters together; even so, these methods usually *still* require ad hoc pre-processing steps (e.g. pre-processing such as Sobel filtering [2, 28]) and extensive hyperparameter tuning. In our method we address this issue and avoid bad local minima in our clustering results by initializing our representation by means of self-supervised learning. In practice, this amounts to train our model on a pretext task (detailed in section 4) and then retain and freeze the earlier layers of the model when applying our clustering algorithm. As reported in [4, 17], the features obtained from self-supervised pre-training are linearly separable with respect to typical semantic image classes. This property is particularly desirable in our context and also motivates our major design choice: since the feature space of self-supervised pre-trained network is linearly separable, it is therefore easier to directly operate on it to discriminate between different clusters.

3.2 Pairwise labeling

A key idea in our method is the choice of space where pairwise the data connections are established: we extract pairwise labels at the level of the data representation rather than at the level of the class predictions. The latter is a common design choice, used in DAC [3] to establish pairwise connections between data points and in DEC [52] to match the current label posterior distribution to a sharper version of itself.

Table 1: **Pairwise labeling with adjacency matrices $A_{ij} = \mathbb{1}_{C_{ij}}$ based on different similarities.** τ is the thresholding hyperparameter for L_2, SNE and Cosine. The number of neighbours k is kNN's hyperparameter.

L_2 dist.	SNE	Cosine	kNN
$C_{ij} = \|\mathbf{f}_j - \mathbf{f}_i\|^2 < \tau$	$\frac{\exp(-\|\mathbf{f}_j - \mathbf{f}_i\|^2/T^2)}{H(Z_i, Z_j)} > \tau$	$\frac{\mathbf{f}_j^\top \mathbf{f}_i}{\|\mathbf{f}_j\|\|\mathbf{f}_i\|} > \tau$	$(j \in \text{kNN}(i)) \vee (i \in \text{kNN}(j))$

The collection of pairwise labels between samples in a mini-batch is given by the adjacency matrix A of an undirected graph whose nodes are the samples and whose edges encode their similarities. DAC [3] generates pseudo labels by checking if the output of the network is above or under certain thresholds. The method of [35] proceeds similarly in the semi-supervised setting. In our method, as we work instead at the feature space level, the pairwise labeling step is a separate process from class prediction and we are free to choose any similarity to establish our adjacency matrix A. We denote with $\mathbf{f}_i \in \mathbb{R}^D$ and $\mathbf{f}_j \in \mathbb{R}^D$ the feature vectors for samples i and j in a mini-batch, obtained from the penultimate layer of the neural network Φ. We also use the symbol $A_{ij} \in \{0, 1\}$ to denote the value of the adjacency matrix for the pair of samples (i, j). Next, we describe the different types of pairwise connections considered in this work and summarize them in table 1.

Cosine and L_2 similarity. Let $\tau \in \mathbb{R}^+$ be a threshold hyperparameter and define $C_{ij} = [\cos(\mathbf{f}_j, \mathbf{f}_i) > \tau]$ (cosine) or $C_{ij} = [\|\mathbf{f}_j - \mathbf{f}_i\|^2 < \tau]$ (Euclidean) where cos denotes the dot product between L_2-normalized vectors. We then define $A_{ij} = \mathbb{1}_{C_{ij}}$ where $\mathbb{1}$ is the indicator function. These definitions connect neighbor samples but do not account well for the local structure of the data. Indeed, it is not obvious that the cosine similarity or Euclidean distance would establish good data connections in feature space.

Symmetric SNE. A possible solution to alleviate the previous issue is to use the symmetric SNE similarity introduced in t-SNE [38]. This similarity is based on the conditional probability $p_{j|i}$ of picking j as neighbor of i under a Gaussian distribution assumption. We make a further assumption compared to [38] of an equal variance for every sample in order to speed up the computation of pairwise similarities and define:

$$p_{j|i} = \frac{\exp(-\|\mathbf{f}_j - \mathbf{f}_i\|^2/T^2)}{\sum_{k \neq i} \exp(-\|\mathbf{f}_k - \mathbf{f}_i\|^2/T^2)} = \frac{\exp(-\|\mathbf{f}_j - \mathbf{f}_i\|^2/T^2)}{Z_i}, \tag{1}$$

$$C_{ij} = \frac{p_{j|i} + p_{i|j}}{2} > \tau \quad \Longleftrightarrow \quad \frac{\exp(-\|\mathbf{f}_j - \mathbf{f}_i\|^2/T^2)}{H(Z_i, Z_j)} > \tau. \tag{2}$$

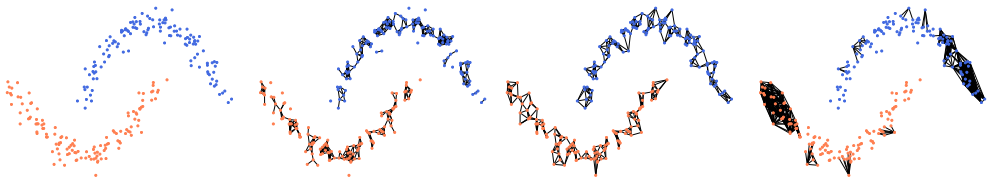
As shown in equation (1), we introduce a temperature hyperparameter $T \in \mathcal{R}^+$ and we call Z_i the partition function for sample i. Then the associated adjacency matrix in equation (2) can be written as a function of the L_2 distance between samples and, in the denominator, of the harmonic mean H of the partition functions. As a result, if sample i or j has many close neighbours, it will reduce the symmetric SNE similarity and possibly prevent a connection between samples i and j. Such a phenomenon is shown on the two moons toy dataset in figure 2.

k-nearest neighbors. We also propose a similarity based on k-nearest neighbours (kNN) [9] where the samples i and j are connected if i is in the k-nearest neighbours of j or if j is in the k-nearest neighbours of i. With this similarity, the hyperparameter is the minimum of neighbours k and not the threshold τ.

3.3 Clustering loss and data composition

Now that we have established pairwise connections between each pair of samples in the mini-batch, we will use the adjacency matrix as target for a binary cross-entropy loss. Denoting with $P(i = j)$ the probability that samples i and j belong to the same cluster, we wish to optimize the clustering loss:

$$L_{\text{clus}} = -\sum_{i,j} A_{ij} \log P(i = j) + (1 - A_{ij}) \log P(i \neq j). \tag{3}$$



(a) Raw data (b) L_2 dist. (c) kNN (d) SNE

Figure 2: **Pairwise connections on the two moons toy data.** From left to right. We apply our algorithm with different connection techniques on a toy dataset shown in (a) where each color represents a class. We use the different connections techniques of table 1 such that there are 650 undirected edges for each similarity. Compared to L_2 distance and SNE, kNN produces neighbourhoods of similar sizes and every sample is connected. SNE captures the local structure of the data: most of the connections are at the external tails of the moons where there are less points.

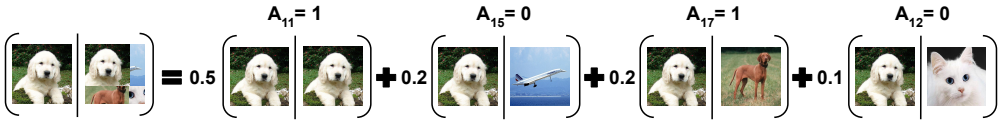


Figure 3: Illustration for equation (6) of a pairwise target between the "pure" image $i = 1$ and the composite image j with $\sigma(j) \in (1, 5, 7, 2)$. In this case, the resulting pairwise target equals 0.7.

The left term of this loss aims at maximizing the number of connected samples (i.e. $A_{ij} = 1$) within a cluster and the right term at minimizing the number of non-connected samples within it (namely, the edges of the complement of the similarity graph $1 - A_{ij} = 1$). Hence the second term prevents the formation of a single, large cluster that would contain all samples.

The next step is to model $P(i = j)$ by using the linear classifier predictions of samples i and j. As seen in equation (4), for a fixed number of clusters K, the probability of samples i and j belonging to the same cluster can be rewritten as a sum of probabilities over the possible clusters. For simplicity, we assume that samples i and j are independent. This way, the pairwise comparison between samples appear only at the loss level and we can thus use the standard forward and backward passes of deep neural networks where each sample is treated independently. By plugging equation (4) in equation (3) and by replacing $\mathbf{p}_j$ with $\mathbf{p}'_j$ to form pairwise comparisons between the mini-batch and its augmented version, we obtain our final clustering loss $\mathcal{L}_{\text{clus}}$:

$$P(i = j) = \sum_{k=1}^{K} P(i = k, j = k) = \sum_{k=1}^{K} P(i = k)P(j = k) = \mathbf{p}_i^\top \mathbf{p}_j, \tag{4}$$

$$L_{\text{clus}} = -\sum_{i,j} A_{ij} \log(\mathbf{p}_i^\top \mathbf{p}'_j) + (1 - A_{ij}) \log(1 - \mathbf{p}_i^\top \mathbf{p}'_j). \tag{5}$$

A similar loss is used in [25] but with supervised pairwise labels to transfer a multi-class classifier across tasks. It is also reminiscent of DAC [3], but differs from the latter because the DAC loss does not contain a dot product between probability vectors but between L_2 normalized probability vectors. Hence DAC optimizes a Bhattacharyya distance whereas we optimize a standard binary cross-entropy loss.

In practice L_{clus} can be used in combination with effective data augmentation techniques such as RICAP [50] and MixUp [56]. These methods combine the images from the minibatch and use a weighted combination of the labels of the original images as new target for the cross-entropy loss. We denote with σ permutation of the samples in the minibatch; RICAP and MixUp require 4 and 2 permutations respectively. RICAP creates a new minibatch of composite images by patching together random crops from the 4 permutations of the original minibatch, whereas MixUp produces a new minibatch by taking a linear combination with random weights from 2 permutations. The new target for a composite image is then obtained by taking a linear combination of the labels in the recombined images, weighted by area proportions in RICAP and the mixing weights in MixUp. These techniques were proposed for the standard supervised classification setting, so we adapt them here to clustering.

In order to do so, we propose to perform a pairwise labeling between the composite images and the raw original images. Both minibatches of original and composite images are fed to the network. Then, as illustrated in figure 3, the pairwise label between a composite image and a raw image is the linear combination of the pairwise labels between the components of both. To sum up, to obtain the pairwise labels between a minibatch and its composite version we just need to extract the adjacency matrix A of the minibatch and then do a linear combination of the adjacency matrix A with the different column permutations σ:

$$L_{\text{clus}} = -\sum_{\sigma} \sum_{i,j} w_{\sigma} A_{i\sigma(j)} \log(\mathbf{p}_i^{\top} \tilde{\mathbf{p}}_j) + (1 - w_{\sigma} A_{i\sigma(j)}) \log(1 - \mathbf{p}_i^{\top} \tilde{\mathbf{p}}_j) \tag{6}$$

Regarding the predicted probability of the 'pure' image i and the composite image j being in the same cluster, we take the dot product between their respective cluster predictions $\mathbf{p}_i$ and $\tilde{\mathbf{p}}_j$.

3.4 Overall loss

The overall loss we optimise is given by

$$L_{\text{tot}} = L_{\text{clus}}(\mathbf{f}, \mathbf{p}, \mathbf{p}') + L_{\text{cons}}(\mathbf{p}, \mathbf{p}'), \tag{7}$$

where

$$L_{\text{cons}} = \frac{\omega(t)}{KN} \sum_{i=1}^{N} \|\mathbf{p}_i - \mathbf{p}'_i\|^2, \tag{8}$$

and $\omega(t) = \lambda e^{-5(1-\frac{t}{T})^2}$ is the ramp-up function proposed in [33, 51] with t the current training step, T the ramp-up length and $\lambda \in \mathbb{R}_+$. L_{cons} is a consistency constraint which requires the model to produce the same prediction $\mathbf{p} \approx \mathbf{p}'$ for an image and an its augmented version. We use it in our method in a similar way as semi-supervised learning techniques [33, 40, 44, 51], i.e. as a regularizer to provide consistent predictions. This differs significantly from clustering methods like IIC [28] and IMSAT [26] where augmentations are used as a main clustering cue by maximizing the mutual information between different versions of an image. Instead, as commonly done in semi-supervised learning, we use the Mean Squared Error (MSE) between predictions as the consistency loss.

4 Experiments

Datasets. We conduct experiments on five popular benchmarks which we use to compare our method against recent state-of-the-art approaches whenever results are available. We use four image datasets and one text dataset to illustrate the versatility of our approach to different types of data. We use MNIST [34], CIFAR 10 [31], CIFAR 100-20 [31] and STL 10 [7] as image datasets. All these datasets cover a wide range of image varieties ranging from 28×28 pixels grey scale digits in MNIST to 96×96 higher resolution images from STL 10. CIFAR 100-20 is redesigned from original CIFAR 100 since we consider only the 20 meta classes for evaluation as common practice [28]. Finally we also evaluate our method on a text dataset, Reuters 10K [36]. Reuters 10K contains 10,000 English news labelled with 4 classes. Each news has 2,000 *tf-idf* features. For all datasets we suppose the number of classes to be known.

Experimental details. We use ResNet-18 [24] for all the datasets except two. For MNIST we use a model inspired from VGG-4 [47], described in [28] and for Reuters 10K we consider a simple DNN of dimension 2000–500–500–2000–4 described in [52]. We train with batch-size of 256 for all experiments. We use SGD optimizer with momentum [49] and weight decay set to 5×10^{-4} for every dataset except for Reuters 10K where we respectively use Adam [30] and decay of 2×10^{-3}. When comparing with other methods in table 2 and table 3, we run our method using 10 different seeds and report average and standard deviation on each dataset to measure the robustness of our method with respect to initialization. As it is common practice [28], we train and test the methods on the whole dataset (this is acceptable given that the method uses no supervision). Further experimental details about data augmentation and training are available in the appendix.

Evaluation metrics. We take the commonly used *clustering accuracy* (ACC) as evaluation metric. ACC is defined as

$$\max_{g \in \text{Sym}(K)} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left\{\overline{y}_i = g\left(y_i\right)\right\}, \tag{9}$$

Table 2: **Comparison with other methods.** Our method almost constantly reaches state-of-the-art performances by a large margin. Note that [28] report best results over all the heads while we report results over ten different initializations. This further shows that our method is overall stable and robust to initialization.

	K-means [39]	JULE [54]	DEC [52]	DAC [3]	IIC [28]	Ours
CIFAR 10	22.9	27.2	30.1	52.2	61.7	**81.7** $\pm$ 0.9
CIFAR 100-20	13.0	13.7	18.5	23.8	25.7	**42.3** $\pm$ 1.0
STL 10	19.2	27.7	35.9	47.0	59.6	**66.4** $\pm$ 3.2
MNIST	57.2	96.4	84.3	97.8	**99.2**	98.6 $\pm$ 0.5

Table 3: **Results on Reuters 10K.** Our method performs on average on par with state of the art. Note that for the best seed we reach state-of-the-art results of 83.5%.

	K-means [39]	IMSAT [26]	DEC [52]	VaDE [29]	FINCH [45]	Ours
Reuters 10K	52.4	71.9	72.2	79.8	**81.5**	79.0 $\pm$ 4.3

where $\overline{y}_i$ and y_i respectively denote the ground-truth class label and the clustering assignment obtained by our method for each sample in the dataset. $\mathrm{Sym}\,(K)$ is the group of permutations with K elements and following other clustering methods we use the Hungarian algorithm [32] to optimize the choice of permutation.

4.1 Results on standard benchmarks

We compare our method with the K-means [39] baseline and recent clustering methods. In table 2, we report results on image datasets. We use RotNet [17] self-supervised pre-training for each dataset on all the data available (*e.g* including the unlabelled set in STL-10). Our method significantly outperforms the others by a large margin. For example, our method achieves 81.5% on CIFAR 10, while the previous state-of-the-art method IIC [28] gives 61.7%. On CIFAR 10, our method also outperforms the leading semi-supervised learning technique FixMatch [48] which obtains 64.3% in its one label per class setting. Similarly, on CIFAR 100-20 and STL 10, our method outperforms other clustering approaches respectively by 14.7 and 6.8 points. On MNIST, our method and IIC both achieve a very low error rate around 1%.

These results clearly show the effectiveness of our approach. Unlike the previous state-of-the-art method IIC that requires to apply Sobel filtering and very large batch size during training, our method does not require such preprocessing and works with a common batch size. We also note that our method is robust to different initialization, with a maximum 3.2% of standard deviation across all datasets.

To analyse further the results on CIFAR 10, we can look at the confusion matrix resulting from our model's predictions.We note that most of the errors are due to the 'cat' and 'dog' classes being confused. If we retain only the confident samples with prediction above 0.9 (around 60% of the samples), the accuracy rises to 94%. We assume that the two classes 'cat' and 'dog' are are more difficult to discriminate due to their visual similarity.

In table 3, we also evaluate our method on the document classification dataset Reuters 10K to show its versatility. We compare with different approaches than in table 2 as clustering methods developed for text are seldom evaluated on image datasets like CIFAR and vice versa. Following existing approaches applied to Reuters 10K, we pretrain the deep neural network by training a denoising autoencoder on the dataset [29]. Our method works notably better than the K-means baseline, and is on par with the best results methods FINCH [45] and VaDE [29]. Most notably one run of our method established state-of-the-art results of 83.5%, 2 points above the current best model.

4.2 Ablation studies

In order to analyze the effects of the different components of our method, we conduct a three parts ablation study on CIFAR 10 and CIFAR 100-20. First, we compare the impact of different possible pairwise labeling methods in the feature space. Second, as one of our key contribution is to choose

Table 4: **Ablation study.** We analyse the effect of different pairwise labeling methods but also the impact of where the labeling is done (feature vs prediction space). We also show the paramount importance of data augmentation for clustering some datasets like CIFAR 10.

	Pairwise labeling				Using the pred. space			Data augmentation		
	L_2	Cosine	kNN	SNE	Cosine	kNN	SNE	RICAP	MixUp	None
CIFAR 10	70.2	81.1	**81.7**	81.5	63.7	64.7	**67.0**	**81.7**	75.3	53.7
CIFAR 100-20	26.1	34.4	**42.3**	40.4	20.4	**32.8**	30.4	**42.3**	37.1	35.4

the space where the pairwise labeling is performed, we test doing so at the level of features and predictions (*i.e.* after the linear classifier but before the softmax layer like DEC [52] or DAC [3]). Third, we analyse the importance of data augmentation in clustering raw images. Results are reported in table 4 and discussed next.

Pairwise similarity. We compare, in feature space, pairwise labeling methods based on L_2 distance, cosine similarity, kNN and symmetric SNE as described in table 1. For kNN, we set the number of neighbors k to 20 and 10 for CIFAR 10 and CIFAR 100-20 respectively. For the cosine similarity, we use respectively thresholds 0.9 and 0.95. For the L_2 distance, we ran a grid search between 0 and 2 to find an optimal threshold. For SNE, we set the threshold to 0.01 and the temperature to 1 and 0.5, for CIFAR 10 and CIFAR 100-20 respectively. Further details about the hyperparameters are available in the supplementary. We observe that kNN, SNE and cosine similarity perform very well on CIFAR 10 with values around 81%. It is interesting to note that cosine similarity performs noticeably worse than kNN and SNE on CIFAR 100-20 with around 6 points less. We also notice that L_2 distance performs consistently worse than the other labeling methods. We can conclude that kNN and SNE are the best labeling methods empirically with consistent performance on these two datasets.

Feature space embedding. Instead of using these labeling methods before the linear classifier, we apply them after it. In this case, our overall approach becomes more similar to standard pseudo-labeling methods such as [3, 35, 52], which aim to match the network predictions output with a 'sharper' version of it. We observe that the performance drops considerably for all labeling methods with an average decrease of 16.3 points for CIFAR 10 and 10.6 points for CIFAR 100-20. Hence, this shows empirically that where pseudo labeling is applied plays a major role in clustering effectiveness and that labeling at the feature space level is noticeably better than doing so at the prediction space level.

Data augmentation. We compare RICAP, MixUp, and the case without data composition (denoted as None). As can it can be seen in table 4, data composition is crucial for CIFAR 10 where RICAP and MixUp surpass None by respectively 28 and 22 points. On CIFAR 100-20, the differences are smaller but using data composition still brings a clear improvement with a 5.1 points increase when using RICAP. Interestingly, RICAP clearly outperforms MixUp in both cases.

5 Conclusions

We have proposed a novel deep clustering method, LSD-C. Our method establishes pairwise connections at the feature space level among different data points in a mini-batch. These on-the-fly pairwise connections are then used as targets by our loss to regroup samples into clusters. In this way, our method can effectively learn feature representation together with the cluster assignment. In addition, we also combine recent self-supervised representation learning with our clustering approach to bootstrap the representation before clustering begins. Finally, we adapt data composition techniques to the pairwise connections setting, resulting in a very large performance boost Our method substantially outperforms existing approaches in various public benchmarks, including CIFAR 10/100-20, STL 10, MNIST and Reuters 10K.

Broader Impact

Our method considers the task of unsupervised clustering from unlabeled data. We mainly consider two types of data: images and text document. While we make significant advances in terms of clustering accuracy compared to previous work, we believe the data we used to be at low risk since we consider datasets wide-spread around the community for sometimes decades.

While the data we used are not at risk we believe there is an inherent risk of misuse with clustering particularly when learnt from raw data. As any learning algorithm the clustering also depends on the data bias and could lead to misinformation or misinterpretation of results obtained from our model.

However we believe our method and clustering in general to be of interest for future years as it would reduce the need of heavy data annotations and processing.

6 Acknowledgments

We thank Kevin Scaman for his very useful comments. This work is supported by the EPSRC Programme Grant Seebibyte EP/M013774/1, Mathworks/DTA DFR02620, and ERC IDIU-638009.

References

[1] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *Proc. ICLR*, 2019.

[2] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proc. ECCV*, 2018.

[3] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *Proc. ICCV*, 2017.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv*, 2020.

[5] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proc. NIPS*, 2016.

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

[7] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011.

[8] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 1979.

[9] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 1967.

[10] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *arXiv*, 2017.

[11] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, 1996.

[12] Dario Fontanel, Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulò, Elisa Ricci, and Barbara Caputo. Boosting deep open world recognition by clustering. *arXiv*, 2020.

[13] Aude Genevay, Gabriel Dulac-Arnold, and Jean-Philippe Vert. Differentiable deep clustering with cluster size constraints. *arXiv*, 2019.

[14] Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *Proc. ICCV*, 2017.

[15] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *Proc. ICCV*, 2019.

[16] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Learning representations by predicting bags of visual words. *arXiv*, 2020.

[17] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *Proc. ICLR*, 2018.

[18] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *IJCAI*, 2017.

[19] Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin. Deep clustering with convolutional autoencoders. In *International conference on neural information processing*, 2017.

[20] Philip Haeusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. Associative deep clustering: Training a classification network with no labels. In *German Conference on Pattern Recognition*, 2018.

[21] Kai Han, Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. Automatically discovering and learning new visual categories with ranking statistics. In *Proc. ICLR*, 2020.

[22] Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *Proc. ICCV*, 2019.

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv*, 2019.

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

[25] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. Multi-class classification without multi-class labels. In *Proc. ICLR*, 2019.

[26] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *Proc. ICML*, 2017.

[27] Gabriel Huang, Hugo Larochelle, and Simon Lacoste-Julien. Centroid networks for few-shot clustering and unsupervised few-shot classification. *arXiv*, 2019.

[28] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proc. ICCV*, 2019.

[29] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. *arXiv*, 2016.

[30] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2014.

[31] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical report*, 2009.

[32] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955.

[33] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *Proc. ICLR*, 2017.

[34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[35] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013.

[36] David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 2004.

[37] Fengfu Li, Hong Qiao, and Bo Zhang. Discriminatively boosted image clustering with fully convolutional auto-encoders. *Pattern Recognition*, 83:161–173, 2018.

[38] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008.

[39] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[40] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Proc. NIPS*, 2019.

[42] Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London Series A*, 1894.

[43] Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Semi-supervised learning with scarce annotations. In *Proc. CVPR Workshop*, 2020.

[44] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in neural information processing systems*, 2016.

[45] Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *Proc. CVPR*, 2019.

[46] Uri Shaham, Kelly Stanton, Henry Li, Boaz Nadler, Ronen Basri, and Yuval Kluger. Spectralnet: Spectral clustering using deep neural networks. In *Proc. ICLR*, 2018.

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014.

[48] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv*, 2020.

[49] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proc. ICML*, 2013.

[50] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *Asian Conference on Machine Learning*, 2018.

[51] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proc. NIPS*, 2017.

[52] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *Proc. ICML*, 2016.

[53] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proc. ICML*. JMLR. org, 2017.

[54] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proc. CVPR*, 2016.

[55] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proc. ICCV*, 2019.

[56] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv*, 2017.

LSD-C: Linearly Separable Deep Clusters Supplementary Material

In this supplementary material, we provide our implementation details, the confusion matrices on CIFAR 10 using our method with kNN labeling and some additional ablation studies. We also include the code to run our method on CIFAR 10 together with the network pretrained with RotNet [17].

A Implementation details

Self-supervised pretraining. We train the RotNet [17] (i.e. predicting the rotation applied to the image among four possibilities: 0°, 90°, 180°, and 270°) on all datasets with the same configuration. Following the authors' released code, we train for 200 epochs using a step-wise learning rate starting at 0.1 which is then divided by 5 at epochs 60, 120, and 160.

Main LSD-C models. After the self-supervised pretraining step, following [21] we freeze the first three macro-blocks of the ResNet-18 [24] as the RotNet training provides robust early filters. We then train the last macro-block and the linear classifier using our clustering method. For all the experiments, we use a batch size of 256. We summarize in table 5 all the hyperparameters for the different datasets and labeling methods.

Table 5: **Hyperparameters.** Optimizer, ramp-up function and parameters of different labeling methods on different datasets.

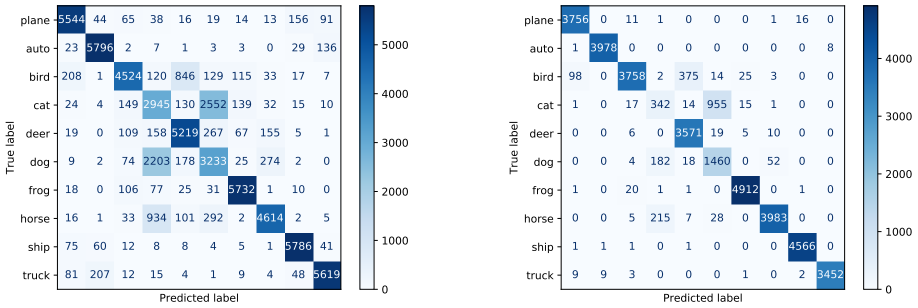
	Optimizer				Ramp-up		Cosine	SNE		kNN
	Type	Epochs	LR steps	LR init	λ	T	τ	τ	Temp	k
CIFAR 10	SGD	220	[140, 180]	0.1	5	100	0.9	0.01	1.0	20
CIFAR 100-20	SGD	200	170	0.1	25	150	0.95	0.01	0.5	10
STL 10	SGD	200	[140, 180]	0.1	5	50	-	0.01	0.5	-
MNIST	SGD	15	-	0.1	5	50	-	-	-	10
Reuters 10K	Adam	75	-	0.001	25	100	-	-	-	5

Data augmentation techniques. We showed in the main paper that data composition techniques like RICAP [50] and MixUp [56] are highly beneficial to our method. For RICAP, we follow the authors' instructions to sample the width and height of crops for each minibatch permutation by using a Beta(0.3, 0.3) distribution. Regarding MixUp, we note that using a Beta(0.3, 0.3) distribution for the mixing weight works better in our case than the Beta(1.0, 1.0) advised for CIFAR 10 in the MixUp paper. Furthermore, we have to decrease the weight decay to 10^{-4} to make MixUp work.

Miscellaneous. Our method is implemented with PyTorch 1.2.0 [41]. Our experiments were run on NVIDIA Tesla M40 GPUs and can run on a single GPU with 12 GB of RAM. Inference time for CIFAR10/CIFAR100-20/STL10 are 10.647s/10.453s/5.517s on a single GeForce GTX 1080 Ti GPU.

B Confusion matrices on CIFAR 10

In figure 4, we show some confusion matrices on CIFAR 10 to analyse how our clustering method performs on the different classes. We notice that there are 8 confident clusters with a very high clustering accuracy of 94.0% for confident samples. The "dog" and "cat" clusters are not well identified possibly due to a huge intra-class variation of the samples.



(a) All samples (b) Confident samples

Figure 4: **Confusion matrices on CIFAR 10 using our method with kNN labeling.** Figure 4a shows that most of the errors are due to the "cat" and "dog" classes. When taking the samples with prediction above 0.9 (60% of the samples) in Figure 4b, there are less than 2000 predictions on classes "cat" and "dog" whereas there are more than 3500 for each of the other classes. Our method manages to ignore the problematic classes when taking the confident samples. Indeed, the accuracy for confident samples is 94.0%.

C Additional experiments

In this section we report results of additional experiments we carried out. In table 6 we evaluate the impact of more components of our method. For example, we apply K-means [39] on the feature space of the pretrained RotNet model and we note very poor performance on CIFAR 10 and CIFAR 100-20. We can conclude that before training with our clustering loss, the desired clusters are not yet separated in the feature space. After training with our clustering loss, the clusters can be successfully separated. Moreover, if we only use the clustering loss and drop the consistency MSE loss, the performance decreases on both CIFAR 10 and CIFAR 100-20 by 1.5 and 1.3 points respectively, showing that the MSE provides a moderate but clear gain to our method. Finally, if we replace the linear classifier by a 2-layer classifier (i.e. this corresponds to a non-linear separation of clusters in the feature space), it results in a small improvement on CIFAR 10 but a clear decrease of 1.9 points on CIFAR 100-20. Hence using a linear classifier provides more consistent results across datasets.

Table 6: **Impact of the different losses.** From the first column, we observe that the desired clusters are not yet separated in the feature space after the RotNet pretraining. The second column shows that the MSE consistency loss provides a boost of more than 1 point to our method. Finally, we see that using a non-linear classifier harms the performance on CIFAR 100-20. All methods were trained for one seed only.

	K-means + RotNet	Ours (kNN)	Ours (kNN) w/o MSE	Ours (kNN) w/ non-lin.
CIFAR 10	14.3	81.7	80.2	82.0
CIFAR 100-20	9.1	40.5	39.2	38.6

Since we only trained the last macro block and the linear layer of a ResNet-18[24] we test the impact of increasing or decreasing the number of trainable weights in table 7. Unsurprisingly, training only the last linear layer reaches about the accuracy of k-means. However we still get reasonable performances by training from the second macro block.

Table 7: **Impact of number of trainable weights.** We test the impact of training the network with our clustering method (SNE) starting from different macro block of ResNet. We report results over two seeds (except #3 which uses 10 seeds). For fairness, we use the same hyper-parameters for each row.

From macro block	#2	#3	#4 (linear layer)
CIFAR 10	53.6±1.4	81.7±0.9	14.7±0.0
CIFAR 100-20	30.6±0.4	42.3±1.0	10.8±0.1

We also study the impact of the self-supervised pretraining method in table 8. We note that our method has less impact when combined with self-supervised training method that are based on contrastive loss. However we note that the data augmentation we used was not the one proposed in the original paper. Hence there might be opportunities for improvements by tweaking hyper parameters or changing the data augmentation.

Table 8: **Impact of self supervised method.** We test the impact of using different self-supervised techniques on CIFAR-10 for one seed (except RotNet which used 10 seeds). All self-supervised method are trained with the same data augmentation. For fairness, all clustering methods use the same hyper-parameters for each row. We trained parameters starting from the third macro block of ResNet-18.

	RotNet[17]	SimCLR[4]	MOCO v2[6]
K-means[39]	14.3	59.45	45.48
Ours (SNE)	81.7	43.5	23.7

Finally since our method uses similarity between elements of a batch, we measure the impact of varying batch size in table 9. We observe different trends in CIFAR-10 and CIFAR100-20. For the former bigger batch sizes are beneficial while this is the opposite for CIFAR100-20. For batch size of 256 the results are stable however. The divergence is probably caused by a higher diversity of data in CIFAR100-20 which results in an unstable training signal.

Table 9: **Impact of batch size.** We test the impact of training with different batch size on our clustering method (SNE). We report results over two seeds (except 256 which used 10 seeds). For fairness, we use the same hyper-parameters for each row.

Batch-size	64	128	256	512
CIFAR 10	72.4±1.3	76.9±1.5	81.7±0.9	80.9±1.2
CIFAR 100-20	46.2±0.2	42.2±0.0	42.3±1.0	31.9±1.0

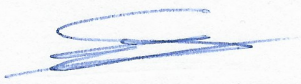
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	LSD-C: Linearly Separable Deep Clusters
Publication Status	□Published □ Accepted for Publication xSubmitted for Publication □Unpublished and unsubmitted work written in a manuscript style
Publication Details	Sylvestre-Alvise Rebuffi*, Sebastien Ehrhardt*, Kai Han*, Andrea Vedaldi and Andrew Zisserman **Submitted** to Advances in Neural Information Processing Systems (2020) *denotes equal contibutions

Student Confirmation

Student Name:	Sebastien Ehrhardt		
Contribution to the Paper	• Contributed to the development of the idea. • Contributed to the development of code library and structure. • Hyper-parameter search on CIFAR-10, CIFAR-100, STL-10 and MNIST. • Evaluation on CIFAR-10, CIFAR-100, STL-10 and MNIST. • Writing paper.		
Signature		Date	27.07.2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Prof. Andrea Vedaldi			
Supervisor comments			
Signature	Andrea Vedaldi	Date	31/07/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 10

Conclusion

Throughout this thesis, we proposed a number of models to make deep learning methods less reliant on human annotations. In the following section 10.1, we will summarise the achievements made by this thesis and the impact it has had on the research community. The last section 10.2 will be dedicated to analyse future developments of our work and possible further directions.

10.1 Achievements and contributions

Long term visual prediction. In chapters 3 and 4, we proposed to perform long term prediction of physical quantities from vision alone. We demonstrated superior results to competitors for scenarios of increasing visual complexity, from synthetic to real data and from one to several objects. The scenarios we proposed went beyond existing benchmarks as the sequences featured motions which were non linear with objects experiencing friction with uneven surfaces. In particular, our work empirically showed that a distributed tensor representation was the best one to perform long term prediction and to handle the uncertainty inherent to this problem. In addition, we showed that combining position and angular velocity prediction resulted in almost consistent improvements for our method as well as for our competitors' methods. Since these quantities are linked by mechanical energy we believe this is a motivation for more physic-based losses when the training signal is complete enough. This was also later highlighted by the work of Greydanus et al. (2019) which incorporated Hamiltonian mechanics constraints into their loss. Another contribution of our work was a simple extension of our propagator to multiple balls. Though simple, this proved to be the most efficient pipeline among others to extrapolate physical quantities of moving objects with various motions.

While chapters 5 and 6 did not solely target long term prediction, we showed that it was possible to learn more interpretable long term predictors from unsupervised data. This was achieved by putting the right inductive priors into the system which was then able to factor different entities and their respective motion to produce realistic video outputs.

Automatic labelling of real data. This thesis also contributed in chapter 4 to an automatic labelling of real data of moving objects through a CNN based tracker. We showed that by using some assumptions on motion continuity and position entropy it was possible to reliably extract positions of one or multiple objects from our unsupervised tracker. This was particularly useful to obtain a training signal for real data in order to train long term predictors. As our dataset is released online, we hope this will encourage practitioners to more often consider real data to train systems that perform intuitive physics predictions.

Objects property inference from past experiences. We also introduced a new meta-learning pipeline in chapter 5 that was trained to make inference from past experiences much like humans can do. One particular achievement of this work was to show that one does not need heavy recurrent neural networks to extract useful information from videos. Instead, past sequences can be efficiently summarised with dynamic images. Inspired from our work on long term prediction we also proposed a distributed representation of the internal physics that allowed to scale over image size, longer time and object number. Finally, we showed that the unconstrained environment representation was modelling important environment properties such as if an object is solid, an occluder or just some background texture.

The technique used to summarise experiences also inspired Groth et al. (2020) to train a VMC that uses dynamic images to summarise the motion history of a robot.

Object-centric generative models. Chapter 6 introduced a novel object-centric model which was able to model interactions in latent space between objects and between objects and their environment. This work significantly differed from all existing prior art since most related works did not enforce such constraint explicitly. When it did, the relation between objects and their environment was not modelled. We quantitatively outperformed all other competitors and, in some scenarios, our model was the only one that managed to produce high fidelity generated images with a correct disentanglement of all objects in the images. Another achievement of this

method was to work on real data and be extended to videos. This allows to make generative models that are able to handle object dynamics in opposite directions which is often badly handled by modern video predictors. While this work has not been published yet, we hope it will encourage research in object-centric generation. Particularly, we hope to inspire research in purely generative object-centric video models which are surprisingly missing in the literature.

Systematic self-supervised pre-training. An important contribution of our work on object recognition in chapters 7 to 9 was to explicitly advocate self-supervised pre-training when annotations are lacking. This impact was not limited to our own algorithm. Indeed, in chapters 7 and 8 we also showed that using self-supervised pre-training for other methods always lead to improvements. Besides, the work presented in this thesis not only showed that this enabled to reach better performances, it also opened a way to more stable training with bigger networks. More particularly, our work in chapter 7 while not yet published at the time, was the first to consider bigger networks, *e.g* ResNet-18 (He et al., 2016) and WideResNet (Zagoruyko & Komodakis, 2016), to perform SSL.

Semi-supervised learning. We showed in chapter 7 that it was possible to train SSL algorithms with very scarce annotations. In particular, starting from a self-supervised representation we performed an alternative optimisation using pseudo-labelling and consistency loss on the unlabelled data. For each set, labelled and unlabelled, we trained different layers to avoid overfitting. As opposed to most recent methods in the field, our algorithm did not rely on heavy data augmentations that are generally only applicable to a particular dataset. We showed that our method was not only effective stand-alone but could also helped to refine representations learnt from semi or fully supervised learning methods. In particular we showed that increasingly better representations lead to better performances of our system.

Even before being published, our work was the first to push the limit of SSL to very scarce annotations. It was also the first to advocate self-labelling in SSL going against some recent studies in SSL (Oliver et al., 2018). The effectiveness of self-labelling was later shown to bring *state-of-the-art* results in SSL benchmarks (Chen et al., 2020b; Sohn et al., 2020).

New category discovery. In chapter 8, we proposed a method that performed automatic new category discovery on standard vision benchmarks. One of the main

innovation of this work was a novel way of comparing two embeddings extracted from a deep neural network. This was achieved by comparing the position of the k biggest values of the embedding for each pair of images. Similar unordered positions were considered to belong to the same category and vice versa. To the best of our knowledge, this was the first time this method was successfully used to compare outputs from a CNN. Another contribution of this work was the use of self-labelling to perform incremental learning. We showed in particular that our method was the one that best disentangled new and old classes. This was highlighted by the incremental learning experiments where in some datasets our method outperformed all other competitors by at least 20 points.

While we only recently submitted the code it has already been starred more than 100 times and been forked 12 times on GitHub which reveals interest of the community.

Deep clustering. Finally, chapter 9 introduced a new technique to perform unsupervised clustering from raw images. By using using pairwise similarity pseudo-labelling of pairs of CNN-embedded images, alongside with a data augmentation technique that combined images as patches, we showed that we could reach almost state-of-the-art results on standard clustering benchmarks. As opposed to concurrent works, we showed that this technique worked on different types of data: visual and text, with different types of self-supervised algorithms. In addition, we show that it is more efficient to compare embeddings extracted from the penultimate layer of a CNN rather than the prediction layer as proposed by other work (Chang et al., 2017a).

10.2 Future directions

Embed video representation with more stochasticity. While we briefly tackled it in chapter 6, but also with the likelihood loss in chapter 3, it is possible to improve our algorithms by predicting the uncertainty of the predictions that are being made. This task is particularly interesting as generating multiple hypotheses seems to be one way to model human perception of the real world (Sanborn et al., 2013). In addition to long term prediction, it would also be interesting to see its potential impact on the interpolation task. This, in order to give possible explanations of an end result from some inputs. Since our work proposed an intermediate tensor representation, directly modelling the uncertainty with a VAE would be the main challenge. In fact, since at the tensor level the activations are spatially correlated,

simply reusing the 'reparametrisation trick' would not be enough. In practice this would be the major limitation to overcome in order to be able to generate multiple hypotheses.

Multiple objects tracking. Another venue for improvements in chapter 4 is to transform the iterative discovery of objects to a single end-to-end training. Since we found that applying our method directly did not work empirically it remains a technical challenge to discover multiple moving objects all at once. A possible way would be to combine optical flow with our method so as to reduce the search space to only moving parts in the field of view.

Deduce more complex environment properties. Chapter 5 introduced a meta-learning pipeline that was able to discover environment properties 'on-the-fly' in 2.5D environment. However, the environment surrounding us is generally 3D and features materials of different properties like friction and restitution coefficient. Hence, we also plan to extend the work carried out in chapter 5 to more complex scenarios. A possible interesting area would be to collect real videos of billiard or pinball plays so as to deduce environment properties (*e.g* billiard orientation, camera angle, mock objects), in order to make future predictions.

RELATE with 3D. In chapter 6, we presented RELATE, an extension of BlockGAN (Nguyen-Phuoc et al., 2020) that modelled interactions between entities of a scene. Our work demonstrated superior performances on 2.5D scenes than the original model but could be extended in two ways. First, an obvious extension of our work would be to model the scenes in 3D as done in the original BlockGAN paper. Some challenges come with it. Modelling the camera pose is a very ambiguous problem. While it can work for scenes with a fixed camera such as CLEVR (Johnson et al., 2017), this becomes an issue for scenarios such as ShapeStacks (Groth et al., 2018) where each scene features a different camera angle. In addition, as of now, BlockGAN only works for relatively low resolution of 64×64 pixels while our work proposed twice the resolution. Porting our current version to model 3D for the same resolution would probably require more work to tweak hyper parameters.

Another potential improvement of RELATE would be to further increase its interpretability. For instance, our method is limited to objects that can be represented by their centroid positions in latent space. Generating individual object segmentation mask would increase our model's capability by tackling more various types of objects.

Potentially this could be done by doing layer-wise composition like in other works (Greff et al., 2019; Anciukevicius et al., 2020) while retaining the spatial stochasticity introduced by BlockGAN at the latent space level.

Object recognition pipeline benchmark. In this thesis, we proposed a pipeline to perform object recognition with less human annotations. This pipeline relied heavily on a first self-supervised training step. While our pipeline was generic, we used RotNet (Gidaris et al., 2018) as the main self-supervision method. This enabled a fair comparison with other methods, simplicity of integration and brought improvements to all methods in SSL and clustering. However, with the recent achievements of self-supervised methods there remain two open questions that our work has not answered yet. The first interesting direction would be a more extensive evaluation of self-supervised methods with SSL and clustering. Using our work or other SSL/clustering methods to build an extensive benchmark of the performances of self-supervised methods would enable researchers to have more clarity on positive/negative interactions between self-supervised methods and SSL/clustering ones. In particular disentangling these methods from the data augmentation techniques they use would give a clearer indication of the broader impact of each method. This type of study was recently approached by Albert et al. (2020) and we hope to see more thorough developments in the future. The other direction would be to systematically evaluate if self-supervision is more effective standalone as a pre-training step than, as Zhai et al. (2019) work suggested, combined with object recognition tasks.

Extension to segmentation and object detection. While in this work we primarily tackled multi-class classification, it is also possible to extend our work to different types of tasks. In particular instance segmentation, object detection and tracking are such tasks where labelling is particularly expensive. Such fields can benefit from our *three-step* pipeline and the various methods we introduced in this thesis in chapters 7 to 9. We note that some recent works in this domain have also tried to port concepts recently used in SSL such as consistency (Jeong et al., 2019) or interpolation (Jeong et al., 2020) which demonstrates that the frontier between the two domains maybe not be very large.

Open questions not limited to the scope of this thesis. Finally we would like to broaden the scope of this thesis by presenting two directions whose impacts are not limited to the tasks or fields introduced in this thesis. First we noted in chapter 6

that StyleGAN-like (Karras et al., 2019) models were not able to handle distributions featuring objects with various scales. While StyleGAN was shown to work on different types of images such as faces or bedrooms, such data often featured similar scales in the scenes. Therefore, it remains an open question whether the conclusion of our work in chapter 6 can be applied to generative methods that use adaptive instance normalisation such as StyleGAN.

A second interesting question brought by this thesis was whether our ranking statistics could be applied to other domains as well. In fact comparing CNN embeddings is not limited to category classification. For instance, in computer vision and in natural language processing, retrieval is generally achieved by comparing outputs of a neural network (Berman et al., 2019). Hence it would be interesting to see whether our ranking statistics procedure can bring substantial improvements to these fields.

Bibliography

Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in neural information processing systems*, 2016.

Paul Albert, Diego Ortego, Eric Arazo, Noel E. O'Connor, and Kevin McGuinness. Relab: Reliable label bootstrapping for semi-supervised learning. *arXiv preprint arXiv:2007.11866*, 2020.

Titas Anciukevicius, Christoph H Lampert, and Paul Henderson. Object-centric image generation with factored depths, locations, and appearances. *arXiv preprint arXiv:2004.00642*, 2020.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. *arXiv preprint arXiv:1908.02983*, 2019.

Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *Advances in neural information processing systems*, 2014.

Renee Baillargeon, Elizabeth S. Spelke, and Stanley Wasserman. Object permanence in five-month-old infants. *Cognition*, 20(3):191–208, 1985.

Anton Bakhtin, Laurens van der Maaten, Justin Johnson, Laura Gustafson, and Ross B. Girshick. Phyre: A new benchmark for physical reasoning. In *Advances in neural information processing systems*, 2019.

Fabien Baradel, Natalia Neverova, Christian Wolf, Julien Mille, and Greg Mori. Object level visual reasoning in videos. In *European conference on computer vision*, 2018.

Fabien Baradel, Natalia Neverova, Julien Mille, Greg Mori, and Christian Wolf. Cophy: Counterfactual learning of physical dynamics. In *International Conference on Learning Representations*, 2020.

Peter W. Battaglia, Jessica Hamrick, and Joshua B. Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 110(45), 2013.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, 2016.

Peter W. Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Maxim Berman, Hervé Jégou, Andrea Vedaldi, Iasonas Kokkinos, and Matthijs Douze. Multigrain: a unified image embedding for classes and instances. *arXiv preprint arXiv:1902.05509*, 2019.

David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *International Conference on Learning Representations*, 2019a.

David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A. Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in neural information processing systems*, 2019b.

Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is nearest neighbor meaningful? In *International conference on database theory*, pp. 217–235. Springer, 1999.

Kiran S. Bhat, Steven M. Seitz, Jovan Popović, and Pradeep K. Khosla. Computing the physical parameters of rigid-body motion from video. In *European conference on computer vision*, 2002.

Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. Action recognition with dynamic image networks. *PAMI*, 40(12), 2017.

Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *International Conference on Machine Learning*, 2001.

John S. Bridle, Anthony JR. Heading, and David JC. MacKay. Unsupervised classifiers, mutual information and'phantom targets. In *Advances in neural information processing systems*, 1992.

Marcus A. Brubaker, Leonid Sigal, and David J. Fleet. Estimating contact dynamics. In *International Conference on Computer Vision*, 2009.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European conference on computer vision*, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *International Conference on Computer Vision*, 2017a.

Michael Chang, Tomer Ullman, Antonio Torralba, and Joshua B. Tenenbaum. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations*, 2017b.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien (eds.). *Semi-Supervised Learning*. MIT Press, 2006.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *International Conference on Machine Learning*, 2020a.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020b.

Yanbei Chen, Xiatian Zhu, and Shaogang Gong. Semi-supervised deep learning with memory. In *European conference on computer vision*, 2018.

Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *International Conference on Computer Vision*, 1997.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, 2006.

Miles Cranmer, Alvaro Sanchez-Gonzalez, Peter W. Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *arXiv preprint arXiv:2006.11287*, 2020.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39, 1976.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Misha Denil, Pulkit Agrawal, Tejas D. Kulkarni, Tom Erez, Peter W. Battaglia, and Nando de Freitas. Learning to perform physics experiments via deep reinforcement learning. In *International Conference on Learning Representations*, 2017.

Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *International Conference on Machine Learning*, 2004.

Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. In *International Conference on Learning Representations*, 2019.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations*, 2019.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

Chelsea Finn, Ian J. Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, 2016a.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016b.

Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. In *International Conference on Learning Representations*, 2016. URL `https://arxiv.org/pdf/1511.07404`.

Fabian B. Fuchs, Adam R. Kosiorek, Li Sun, Oiwi Parker Jones, and Ingmar Posner. End-to-end recurrent multi-object tracking and trajectory prediction with relational reasoning. *arXiv preprint arXiv:1907.12887*, 2019.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36, 1980.

Felix A. Gers and Jürgen Schmidhuber. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Trans. on Neural Networks*, 12(6), 2001.

Kamran Ghasedi Dizaji, Amirhossein Herandi, Cheng Deng, Weidong Cai, and Heng Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *International Conference on Computer Vision*, 2017.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

Rohit Girdhar and Deva Ramanan. Cater: A diagnostic dataset for compositional actions & temporal reasoning. In *International Conference on Learning Representations*, 2020.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, 2005.

Klaus Greff, Raphaël Lopez Kaufmann, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, 2019.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in neural information processing systems*, 2019.

Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *European conference on computer vision*, 2018.

Oliver Groth, Chia-Man Hung, Andrea Vedaldi, and Ingmar Posner. Goal-conditioned end-to-end visuomotor control for versatile skill primitives. *arXiv preprint arXiv:2003.08854*, 2020.

Radek Grzeszczuk, Demetri Terzopoulos, and Geoffrey E. Hinton. Neuroanimator: Fast neural network emulation and control of physics-based models. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 9–20, 1998.

Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Proceeding of the International Joint Conference on Artificial Intelligence*, 2017.

Abhinav Gupta, Alexei A. Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *European conference on computer vision*, 2010.

Philip Haeusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. Associative deep clustering: Training a classification network with no labels. In *German Conference on Pattern Recognition*, 2018.

Jessica B. Hamrick. Internal physics models guide probabilistic judgments about object dynamics. In *Proceedings of the 33rd annual conference of the cognitive science society*, 2011.

Jessica B. Hamrick, Peter W. Battaglia, Thomas L. Griffiths, and Joshua B. Tenenbaum. Inferring mass in complex scenes by mental simulation. *Cognition*, 157, 2016.

Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *International Conference on Computer Vision*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *International Conference on Computer Vision*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *International Conference on Computer Vision*, 2017.

Susan J. Hespos and Kristy VanMarle. Physics for infants: Characterizing the origins of knowledge about objects, substances, and number. *Wiley Interdisciplinary Reviews: Cognitive Science*, 3(1):19–27, 2012.

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

Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks. In *International Conference on Learning Representations*, 2018.

Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. Multi-class classification without multi-class labels. In *International Conference on Learning Representations*, 2019.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International Conference on Machine Learning*, 2017.

Drew A. Hudson and Christopher D. Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations*, 2018.

Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Jacob Jackson and John Schulman. Semi-supervised learning by label gradient alignment. *arXiv preprint arXiv:1902.02336*, 2019.

Michael Janner, Sergey Levine, William T. Freeman, Joshua B. Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-oriented prediction and planning. In *International Conference on Learning Representations*, 2018.

Miguel Jaques, Michael Burke, and Timothy M. Hospedales. Physics-as-inverse-graphics: Joint unsupervised learning of objects and physics from video. In *International Conference on Learning Representations*, 2020.

Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. In *Advances in neural information processing systems*, 2019.

Jisoo Jeong, Vikas Verma, Minsung Hyun, Juho Kannala, and Nojun Kwak. Interpolation-based semi-supervised learning for object detection. *arXiv preprint arXiv:2006.02158*, 2020.

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *International Conference on Computer Vision*, 2019.

Zhaoyin Jia, Andrew Gallagher, Ashutosh Saxena, and Tsuhan Chen. 3d-based reasoning with blocks, support, and stability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: an unsupervised and generative approach to clustering. In *Proceeding of the International Joint Conference on Artificial Intelligence*, 2017.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross B. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Ken Kansky, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *International Conference on Machine Learning*, 2017.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

In-Kyeong Kim and Elizabeth S. Spelke. Perception and understanding of effects of gravity and inertia on object motion. *Developmental Science*, 2(3):339–362, 1999.

Durk P. Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Durk P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, 2014.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, 2018.

Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. In *Advances in neural information processing systems*, 2018.

Jannik Kossen, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting. Structured object-aware physics prediction for video modeling and planning. In *International Conference on Learning Representations*, 2019.

Andreas Krause, Pietro Perona, and Ryan G. Gomes. Discriminative clustering by regularized information maximization. In *Advances in neural information processing systems*, 2010.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

James R. Kubricht, Keith J. Holyoak, and Hongjing Lu. Intuitive physics: Current research and controversies. *Trends in cognitive sciences*, 21(10):749–759, 2017.

Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli, and Joshua B. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in neural information processing systems*, 2015.

Lubor Ladicky, SoHyeon Jeong, Barbara Solenthaler, Marc Pollefeys, and Markus H. Gross. Data-driven fluid simulations using regression forests. *ACM Trans. Graph.*, 2015.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266), 2015.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng. Building high-level features using large scale unsupervised learning. *International Conference on Machine Learning*, 2012.

Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2003.

Adam Lerer, Sam Gross, and Rob Fergus. Learning physical intuition of block towers by example. In *International Conference on Machine Learning*, 2016.

Wenbin Li, Aleš Leonardis, and Mario Fritz. Visual stability prediction and its application to manipulation. In *AAAI conference on artificial intelligence*, 2017.

Yunzhu Li, Antonio Torralba, Animashree Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. *arXiv preprint arXiv:2007.00631*, 2020.

Zhijian Liu, William T. Freeman, Joshua B. Tenenbaum, and Jiajun Wu. Physical primitive decomposition. In *European conference on computer vision*, 2018.

David Lopez-Paz, Krikamol Muandet, Bernhard Schölkopf, and Iliya Tolstikhin. Towards a learning theory of cause-effect inference. In *International Conference on Machine Learning*, 2015.

David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Scholkopf, and Léon Bottou. Discovering causal signals in images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Michael Lutter, Christian Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2018.

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. In *International Conference on Machine Learning*, 2016.

James MacQeen. Some methods for classification and analysis of multivariate observations. In *Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, 1967.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *European conference on computer vision*, 2018.

Michael McCloskey, Allyson Washburn, and Linda Felch. Intuitive physics: The straight-down belief and its origin. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 9(4):636, 1983.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *PAMI*, 41(8), 2019.

Aron Monszpart, Nils Thuerey, and Niloy J. Mitra. SMASH: physics-guided reconstruction of collisions from videos. *ACM Trans. on Graphics (TOG)*, 2016.

Joris M. Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102, 2016. URL `http://jmlr.org/papers/v17/14-518.html`.

Roozbeh Mottaghi, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016a.

Roozbeh Mottaghi, Mohammad Rastegari, Abhinav Gupta, and Ali Farhadi. what happens if... learning to predict the effect of forces in images. In *European conference on computer vision*, 2016b.

Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li F Fei-Fei, Joshua B. Tenenbaum, and Daniel L. Yamins. Flexible neural representation for physics prediction. In *Advances in neural information processing systems*, 2018.

Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, 2001.

Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv preprint arXiv:2002.08988*, 2020.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, 2016.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder P. Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, 2015.

Avital Oliver, Augustus Odena, Colin Raffel, Ekin Dogus Cubuk, and Ian J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in neural information processing systems*, 2018.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Xi Peng, Shijie Xiao, Jiashi Feng, Wei-Yun Yau, and Zhang Yi. Deep subspace clustering with sparsity prior. In *Proceeding of the International Joint Conference on Artificial Intelligence*, 2016.

Tu-Hoa Pham, Abderrahmane Kheddar, Ammar Qammaz, and Antonis A Argyros. Towards force sensing from vision: Observing hand-object interactions to infer manipulation forces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Luis Piloto, Ari Weinstein, Dhruva TB, Arun Ahuja, Mehdi Mirza, Greg Wayne, David Amos, Chia-chun Hung, and Matt Botvinick. Probing physics knowledge using tools from developmental psychology. *arXiv preprint arXiv:1804.01128*, 2018.

Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *European conference on computer vision*, 2016.

Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. Deep co-training for semi-supervised image recognition. In *European conference on computer vision*, 2018.

Marc'Aurelio Ranzato, Fu Jie Huang, Y-Lan Boureau, and Yann LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*, 2015.

Davis Rempe, Srinath Sridhar, He Wang, and Leonidas Guibas. Predicting the physical dynamics of unseen 3d objects. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 2834–2843, 2020.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *International Conference on Machine Learning*, 2011.

Ronan Riochet, Mario Ynocente Castro, Mathieu Bernard, Adam Lerer, Rob Fergus, Véronique Izard, and Emmanuel Dupoux. Intphys: A framework and benchmark for visual intuitive physics reasoning. *arXiv preprint arXiv:1803.07616*, 2018.

Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. In *Proceedings of the Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTION'05)-Volume 1-Volume 01*, 2005.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in neural information processing systems*, 2016.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in neural information processing systems*, 2016.

Adam N. Sanborn, Vikash K. Mansinghka, and Thomas L. Griffiths. Reconciling intuitive physics and newtonian mechanics for colliding objects. *Psychological review*, 120(2):411, 2013.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin A Riedmiller, Raia Hadsell, and Peter W. Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *arXiv preprint arXiv:2002.09405*, 2020.

Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, 2017.

Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On causal and anticausal learning. In *International Conference on Machine Learning*, 2012.

Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European conference on computer vision*, 2012.

David Silver, Hado Hasselt, Matteo Hessel, Tom Schaul, Arthur Guez, Tim Harley, Gabriel Dulac-Arnold, David Reichert, Neil Rabinowitz, Andre Barreto, et al. The predictron: End-to-end learning and planning. In *International Conference on Machine Learning*, 2017.

Kevin Smith, Lingjie Mei, Shunyu Yao, Jiajun Wu, Elizabeth S. Spelke, Joshua B. Tenenbaum, and Tomer Ullman. Modeling expectation violation in intuitive physics with coarse probabilistic object representations. In *Advances in neural information processing systems*, 2019.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *International Conference on Learning Representations*, 2016.

Trevor Standley, Ozan Sener, Dawn Chen, and Silvio Savarese. image2mass: Estimating the mass of an object from its image. In *Conference on Robot Learning*, 2017.

Russell Stewart and Stefano Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *AAAI conference on artificial intelligence*, 2017.

Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advances in neural information processing systems*, 2013.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL `http://arxiv.org/abs/1312.6199`.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, 2017.

James Thewlis, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object frames by dense equivariant image labelling. In *Advances in neural information processing systems*, 2017.

Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. Learning deep representations for graph clustering. In *AAAI conference on artificial intelligence*, 2014.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating eulerian fluid simulation with convolutional networks. In *International Conference on Machine Learning*, 2017.

Lars Van der Maaten and Geoffrey E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 2008.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Learning to classify images without labels. In *European conference on computer vision*, 2020.

Sjoerd van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. In *International Conference on Learning Representations*, 2018.

Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. In *Proceeding of the International Joint Conference on Artificial Intelligence*, 2019.

Zhihua Wang, Stefano Rosa, Yishu Miao, Zihang Lai, Linhai Xie, Andrew Markham, and Niki Trigoni. Neural allocentric intuitive physics prediction from real videos. *arXiv preprint arXiv:1809.03330*, 2018.

Nicholas Watters, Andrea Tacchetti, Theophane Weber, Razvan Pascanu, Peter W. Battaglia, and Daniel Zoran. Visual interaction networks. *arXiv.cs*, abs/1706.01433, 2017.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural networks: Tricks of the trade*, pp. 639–655. Springer, 2012.

Terry Winograd. Understanding natural language. *Cognitive psychology*, 3(1):1–191, 1972.

Jiajun Wu, Ilker Yildirim, Joseph J. Lim, Willam T. Freeman, and Joshua B. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in neural information processing systems*, 2015.

Jiajun Wu, Joseph J. Lim, Hongyi Zhang, Joshua B. Tenenbaum, and William T. Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *Proc. BMVC*, 2016a.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in neural information processing systems*, 2016b.

Jiajun Wu, Erika Lu, Pushmeet Kohli, William T. Freeman, and Joshua B. Tenenbaum. Learning to see physics via visual de-animation. In *Advances in neural information processing systems*, 2017.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, 2016.

Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *International Conference on Machine Learning*, 2017.

Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016a.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016b.

Tian Ye, Xiaolong Wang, James Davidson, and Abhinav Gupta. Interpretable intuitive physics model. In *European conference on computer vision*, 2018.

Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani. Compositional video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. Clevrer: Collision events for video representation and reasoning. In *International Conference on Learning Representations*, 2020.

Kuan-Ting Yu, Maria Bauzá, Nima Fazeli, and Alberto Rodriguez. More than a million ways to be pushed. A high-fidelity experimental dataset of planar pushing. In *Proc.IROS*, 2016.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv.cs*, abs/1605.07146, 2016.

Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst D. Simon. Spectral relaxation for k-means clustering. In *Advances in neural information processing systems*, 2002.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Renqiao Zhang, Jiajun Wu, Chengkai Zhang, William T. Freeman, and Joshua B. Tenenbaum. A comparative evaluation of approximate probabilistic simulation and deep neural networks as accounts of human physical scene understanding. In *CogSci*, 2016.

Bo Zheng, Yibiao Zhao, Joey Yu, Katsushi Ikeuchi, and Song-Chun Zhu. Scene understanding by reasoning stability and safety. In *International Conference on Computer Vision*, 2015.

David Zheng, Vinson Luo, Jiajun Wu, and Joshua B. Tenenbaum. Unsupervised learning of latent physical properties using perception-prediction networks. *arXiv preprint arXiv:1807.09244*, 2018.

Hong-Yu Zhou, Avital Oliver, Jianxin Wu, and Yefeng Zheng. When semi-supervised learning meets transfer learning: Training strategies, models and datasets. *arXiv.cs*, abs/1812.05313, 2018.

Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine Learning*, 2003.

Yixin Zhu, Chenfanfu Jiang, Yibiao Zhao, Demetri Terzopoulos, and Song-Chun Zhu. Inferring forces and learning human utilities from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.