BEYOND VANILLA FINETUNING: APPROACHES TO MAXIMIZE THE BENEFITS OF PRETRAINING IN COMPUTER VISION

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Bram Wallace

December 2021

BEYOND VANILLA FINETUNING: APPROACHES TO MAXIMIZE THE BENEFITS OF PRETRAINING IN COMPUTER VISION

Bram Wallace, Ph.D.

Cornell University 2021

The field of computer vision has benefited tremendously from an unusual blessing: a baseline that works quite well on almost every problem, is extremely simple to implement, and sometimes shockingly hard to surpass. The existence of such a baseline runs at odds to the history of vision; prior to deep learning, approaches historically required careful design to obtain non-trivial performance on many benchmarks. This baseline is the process of pretraining and finetuning; initially training a neural network on a large dataset to perform some task and then applying components of that network to new tasks with minimal modifications or additions. Specifically, the feature extractor (dimensionality-reducing module) of the initial network is re-used with only a small task-specific output network or layer needing to be trained from scratch. In the last decade, this simple process has served as a building block for state-of-the-art on numerous vision benchmarks, sometimes with minimal to no modifications.

For all of its success, however, this method is still understudied and underdeveloped. Until recently, there was very little effort to determine *which* task to pretrain this network on; supervised training on the ImageNet classification benchmark was simply considered sufficient. There has been some study about *why* this transfer occurs, but relatively few attempts to *better* the transfer, especially from a methodology instead of parameter optimization perspective. In

this thesis, I consider from multiple perspectives the problem of using prior knowledge and pretraining when given a new collection of images.

Chapter 1 introduces the method of pretraining and finetuning in further detail and gives a summary of the contributions of this work.

In Chapter 2, I consider a variety of self-supervised learning methods on an extremely varied set of domains, with the goal of understanding *what signals exist to learn* and what signals are exploited or unexploited by pretraining on or off the domain. I also consider how the pretraining and finetuning process compares to domain-specific self-supervised learning. This work was published in ECCV 2020.

In Chapter 3, we address the problem of improving the feature extraction process directly. A novel architecture and algorithm for the self-supervised ensembling of pretrained networks on a novel dataset is presented. This method dramatically improves nearest-neighbor classification performance on generalized-to datasets, and even is applicable to the single-model (non-ensemble) setting. This work was performed during an internship at Salesforce Research under the mentorship of Devansh Arpit and is currently under review.

In Chapter 4, we consider the improvement of transfer learning by domain-specific pretraining. The considered task, called *expert selection*, deals with choosing the optimal pretraining category out of a given set. Here we match the previous state-of-the-art's fully supervised performance despite our method not requiring labels on the target or source domains. Additionally, we present a more general form of the method which requires weaker assumptions about the existence of powerful pretrained networks. This work was part of a collaboration with Ziyang Wu during his time as a Master's student at Cornell and was published in CVPR 2021.

Chapter 5 provides a change of pace, focusing instead on the problem of task-specific *generalization*. Specifically, the problem of inter-class generalization is considered in the context of monocular 3D reconstruction. An architecture is presented that utilizes an inputted prior at inference time to generalize across class categories with no retraining required, only an exemplar shape from the new class. This work was published in ICCV 2019.

Chapter 6 brings the technical contributions to a close. Like the previous chapter, here we move away from the problem of image recognition. Specifically, timeseries analysis and classification is considered. We present in-progress work to build a generalizable model for timeseries that parallels the at-times seemingly omniscient power of pretrained vision and language models. Concretely, we demonstrate a currently per-dataset deep metric learning architecture that exceeds the unsupervised state of the art for a wide benchmark of timeseries classification tasks. This work will be continued in the future. The presented initial progress was made while working with Kavita Bala and Samar Khanna during his undergraduate studies at Cornell.

Finally, Chapter 7 discusses future directions of both the presented work and field at large.

BIOGRAPHICAL SKETCH

Bram was born to Deborah and Brett Wallace on January 24, 1996. He grew up in San Mateo, California, where he was homeschooled by his mother and a large variety of books. He took advantage of this extra freedom by focusing on math, as well as on other important topics such as taekwondo, soccer, Dungeons & Dragons, Pokémon cards, and building rope swings. In 2010, Bram enrolled at the local junior college, College of San Mateo, where he continued to focus on mathematics while competing for the cross-country and track team. In 2013, he transferred to UC Santa Barbara where he majored in applied math as well as scheduling optimization problems; particularly the problem of how to organize his class and homework schedule to maximize the number of 3 and 4 day weekends in his life. As part of his senior year studies, he began coding for numerical analysis which led to conducting research with Paul Atzberger following Bram's graduation in Spring 2015. In Fall 2015, by the extreme graciousness of his parents, he moved back to his boyhood home in San Mateo where he extensively developed his programming skills while deciding on his next steps. In Spring 2016, these next steps materialized as year-long internship at a startup, Vium. During the internship he applied to PhD programs, and ultimately elected to return to academia at Cornell (didn't see that one coming). Very importantly, on October 23rd, 2017, Bram met Melanie Maurer in a rock climbing class. This event took on increasing consequence as they began dating in the winter of 2018 and got engaged on May 21st, 2021. At the beginning of his second year at Cornell, following a summer internship at Bloomberg LP in financial engineering, Bram began work in computer vision while advised by Bharath Hariharan. After graduation, he will be joining Salesforce Research's AI for Society group where he looks forward to conducting a mixture of funda-

mental and philanthropic machine learning research.

This thesis is dedicated to my mom and dad. Thanks for being just as good of friends as you are parents.

ACKNOWLEDGEMENTS

I am deeply grateful to the many people who have helped and supported me through the period of my life that has culminated in this thesis. First, I thank Bharath Hariharan. As my advisor, no one else has been as important in my academic growth these last several years as him. In particular, I am incredibly appreciative of accepting me as a student coming from the Center for Applied Mathematics instead of his home Computer Science department; this opportunity shaped both my time at Cornell and the opportunities that now follow. The trust that Bharath placed in both my research adventures and misadventures, through both highlights and lowlights, was invaluable. The brainstorming sessions we've had were both incredibly fun, as well as extremely formative in my research approach and outlook.

Next, I thank others who have had a hand in my academic growth these last few years. Beginning with the Center for Applied Mathematics: I thank the director's Alex Vladimirsky and David Bindel for their leadership of the program, as well as the administrative head Erika Fowler-Decatur for being the shepherd and glue of CAM during the vast majority of my time there. David Eriksson's advice as a senior student/alumni throughout my PhD has been irreplaceable, as well as his eagerness to always hike a few extra miles. From my unofficially adopted department of Computer Science, I would like to first thank Geoff Pleiss for giving me the opportunity to work with him in the very beginning stages of my PhD as well the numerous opportunities to go climb cliffs with him towards the end of my PhD. Next, I am grateful to my research group for being the first sounding board for new ideas, providing solidarity during deadline pushes, and bringing me up to speed on the ins and outs of computer vision when I was first joining. In particular, I thank Davis Wertheimer for our excel-

lent mathematical discussions, Cheng Perng Phoo for his deep technical knowledge of machine learning, Luming Tang for his enthusiasm and out-of-the-box thinking, Utkarsh Mall for his even keel and interesting projects, Ziyang Wu for bringing Neural Tangent Kernel literature to my attention, and Guandao Yang for his unbelievable ability to keep both himself and anyone who talks to him abreast of literature. Finally, I thank my committee members Marten Wegkamp and Peter Frazier for their guidance.

On the personal side, I would not be where I am today without my parents. From guiding my education early on (both academically as well as sparking my love of exercise and running), to figuring out how I could legally begin junior college young, to letting me live at home for over two years between UCSB and Cornell, to helping me edit my PhD application writing, to hosting me researching remotely during my PhD, there are more ways that they have lifted me up and supported me over the years than I can count.

Getting a PhD is rarely easy, and this one was no exception. I am lucky, however, to have a set of incredible friends around me who have helped make the last few years downright enjoyable. As previously mentioned, both David Eriksson and Geoff Pleiss are steadfast friends who I've enjoyed many outdoor escapades with. Joseph Long has known me and Melanie for as long as we've known each other; I'm still not sure if he meant to wingman us but I'm sure glad that it worked out that way. Richa Agrawal's optimism and energy have buoyed us all through the more frustrating PhD times, and we all owe her for the hours of video games played at her apartment. No friend has been more of a constant presence during my time at Cornell than Misha Padidar: thanks for all the jokes, workouts, rants, research sessions, celebrations, and motivation given in an Arnold Schwarzenegger accent (as well as establishing the CAM Gaucho

legacy).

Finally, and most important of all, I thank Melanie Maurer. From best friend during our first semester, to girlfriend from our second onwards, to fiancée now, you have been as defining a part of my life these last few years as my studies. You are a constant inspiration, an unwavering support, an endless source of fun, and the love of my life. Thank you for everything, now and always.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

To begin this thesis with the now-classic echo (up to paraphrasing) that "the application of deep learning to computer vision has enjoyed rapid growth in recent years in terms of both performance and interest", would be a massive understatement. Few fields have benefited from the advent of neural networks and graphics processing units as computer vision has. Possibly only natural language processing has reaped comparable benefits.

In what ways have these benefits been realized? What tasks have been pushed forward? From an academic perspective, there are dozens of benchmarks and methods of measure on said benchmarks that have been proposed and utilized. Tasks that one might perform include classification, multi-class classification, open-set classification, semantic segmentation, depth estimation, inpainting, image generation, 3-D reconstruction, object tracking, shape completion, or novel viewpoint synthesis. These tasks can be performed with varying amounts of data and label availability, a sampling of such settings include full supervision, semi-supervision, self-supervision, weak supervision, one-shot learning, few-shot learning, or zero-shot learning. A third axis that problem settings vary in is the domain itself, i.e. the type and content of the imagery. A diverse but far from exhaustive sampling of image domains is employed in Chapter 2. Here it suffices to say that in addition to the power set of things (in literally any definition of the word "thing") that human beings have visually observed since cameras were invented there is a myriad of non-human-visible visual signal capturing techniques such as medical scans, microscopy, aerial imaging, astronomical telescopes, and more that can provide the broad

category of data that we call images. This is not to mention performing vision on things that don't even exist, such as the synthetic data used in SYNTHIA[160]. So the short answer to the leading questions of this paragraph might be "many" and "lots".

Throughout this overwhelming diversity of problem settings that deep learning has been applied to, however, certain things remain stable and constant. As an example (which this thesis will not discuss at length), stochastic gradient descent (SGD) is one of the only successful ways to train a neural networks and indeed the *only* way that has shown widespread success. Such is of course by design, the differentiability of neural networks is a core tenet of their architecture. A more unexpected constant theme, is the astoundingly widespread efficacy of a baseline known as a pretrained ImageNet feature extractor.

While progress is often measured academically by the optimal performance that carefully crafted and tailored methods can achieve, perhaps a better measure of the practicality of a field is the strength, utility, and reliability yielded by a core subset of approaches. If such a toolbox exists, is easy to implement, and yields strong results on the vast majority of in-the-wild problems, then it would seem that the aforementioned field is quite poised for rapid adoption and application. For computer vision, this has indeed been the case. In fact, computer vision has as even stronger version of this property; taking a pretrained ImageNet feature extractor (which will be described precisely momentarily) and finetuning it with whatever labels are available on the desired downstream task, yields a bar of performance on many academic benchmarks that is often extremely challenging to overcome, or even historically insurmountable. For many tasks,

this efficacy of pretraining is accepted and fully leveraged: many early deep learning approaches that achieved state-of-the-art results on problems such as object detection simply utilized the power of pretrained networks adapted to specific tasks. Even recently, these pretrained feature extractors have been used as the backbone for many more refined state-of-the-art approaches, with the network initialization serving as a critical component to beat simple transfer learning baselines. So the story of deep learning in computer vision up until now is untellable without the inclusion of the pretrained ImageNet feature extractor. I now explain exactly what this baseline is and how it is used in practice.

1.1 Pretrain on ImageNet, Finetune Anywhere

One of the key obstacles in computer vision is the high dimensionality of the data. Even simple 28×28 grayscale images have basic representations of nearly 800 dimensions, a similarly sized image with the 3 typical RGB channels has triple that. Most images, however, have pixels on the order of tens of thousands. This is one of the defining features of computer vision: the dimensionality is so large that many methods are rendered impractical. In addition to the computational constraints, the nature of imagery means that flat vectors of representations are very decorrelated with semantic meaning. Scaling, shifting, or rotating an image will dramatically alter the pixel values even though the *semantics* (meaning) and geometry of an image will be largely preserved. This leads to neural networks for images typically devoting the bulk of the architecture to mapping their input to a vector or spatial grid of tractable size for which the values contain information about attributes that are meaningful for the problem at hand. This process is called *feature extraction* and this large initial segment of

the network is referred to as a *feature extractor* (denoted as Θ for the purpose of Chapter 1).

Once feature extraction has been performed on an image, the new, typically now much smaller, representation is then fed into a *head* network (denoted as ϕ for the purpose of Chapter 1) which computes the output (i.e. $output = \phi(\Theta(input))$. For a task such as classification, this head network is often just a simple linear classifier, or possibly a multi-layer perceptron (MLP), trained on top of a vector representation. For more complicated visual tasks such as detection or segmentation, the feature will typically be a spatial map of reduced dimensionality and large number of feature channels (analogous to the original RGB values) and the head network will have more sophisticated (de)convolutional layers.

A key motivation behind this process is that if the feature representation is suited enough to the task at hand then simple methods (or even classical ones like k-nearest neighbors) can provide good performance when trained on these features. One might initially suppose that losslessly capturing the data of the image in the feature representation might be sufficient, but this is not the case. As discussed in Chapter 2 such a process, called autoencoding, does not produce a well-behaved enough representation space in comparison to more intentionally crafted spaces, this is partly attributable to the fact that even the total of captured imagery in the world is extremely sparse in pixel space, breaking the guarantees of classical methods. The most successful feature extractors are those that are trained in alignment with the goal of the head network. For example, a classification network with a linear head ϕ will simultaneously have Θ trained resulting in a linearly separable feature space. For this reason, the pa-

rameters of Θ are often tuned in conjunction with the training of ϕ. This allows for a more expressive function to be learned, and means that pretraining can be leveraged with practically zero sacrifice.

A very important classification task commonly used for pretraining is the ImageNet Large Scale Visual Recognition Challenge (although nowadays it is considered a moderate-scale problem). This challenge is performed across a dataset of 1.3 million images and 1000 classes, which primarily consist of a variety of animals, food, and everyday objects. ImageNet classification was a grand challenge in computer vision for a long time, success on this benchmark using a convolutional neural network is widely cited as the impetus for the explosion of deep learning. Such a network is trained in the manner described above. A feature extractor, Θ, and an (often linear) head, ϕ, are jointly trained via stochastic gradient descent to map an image x_i that belongs to class y_i to the one-hot vector $\vec{e}_{y_i}$ as guided by a cross-entropy loss. This approach, in various iterations and modifications to algorithm and architecture, has steadily yielded improved performance on the ImageNet benchmark.

The exciting part of the story, however, comes when the feature extractor of such a network, denoted here as Θ_0, is applied to new datasets and types of imagery. Despite being trained on a specific type of imagery (namely object-centric photographs uploaded to the internet) over a specific set of 1000 classes (which albeit large is still extremely narrow in scope from a global perspective), the representations that Θ_0 yields is incredibly useful across a staggering variety of uses and imagery domains. For classification datasets ranging from satellite imagery, to medical scans, to hand-written symbols, and more, the features provided by Θ_0 have been shown to provide excellent performance under even

simple classification methods such as linear regression or k-nearest neighbors. More generally, when given a new computer vision problem, simply using a Θ_0 feature extractor combined with a trained-from-scratch linear head ϕ' typically yields extremely competitive performance.

In essence, the existence of such easily obtainable Θ_0 bypasses the major problem of high dimensionality in images by reducing nearly arbitrary imagery into a parseable space. Perhaps more than anything else, this phenomenon is what has spurred the rapid proliferation of vision applications. The author even found this approach useful when binarily detecting the presence of lab mice on an "exercise ladder" from just a few hundred training examples while interning at a biotechnology startup in 2016.

This generalizability of networks pretrained on large-scale tasks is a key vision property that is leveraged throughout both literature and application. There is some existing work on the refinement of this process: billion-scale datasets with more classes yield even more generalizable representations[120] and training on ImageNet in different ways (some not even using the provided labels) similarly provides a Θ_0 with even more empirical transfer efficacy[76, 29, 236]. This process of transferring a representation, called *transfer learning*, is a whole sub-field of computer vision itself, with work such as [99] specifically optimizing the finetuning of a Θ_0 in conjunction with the training of a new head ϕ. I lay out my contributions to the understanding and improvement of the use of pretraining and pretrained feature extractors, in the following section.

1.2 Contributions

The goal of my research has been to expand and improve the method of pretraining and transfer learning. In Chapter 2, the benefit of self-supervised (no label) learning is studied on a wide array of domains in order to investigate *what* knowledge and techniques are useful in both specific and general instances. Domain-specific advantages to self-supervised learning techniques are uncovered that inform the rest of my investigations as well as identifying general trends across all domains. Self-supervised learning can offer competitive advantages over or in addition to ImageNet finetuning as *unlabeled* images are learned from. In Chapter 3, a method to improve the representations of either a single or multiple ensembled Θ_0 on a given domain is presented. This contribution offers an out-of-the-box approach to improve pretrained feature extractor representations given a task, either individually or in an ensemble (which is a novel setting). In Chapter 4, domain-specific pretraining is explored: pretraining on a broad diverse set of images is less effective than training on a smaller but extremely relevant set. The task of expert selection is tackled in the zero-label and zero-initialization contexts with results matching the supervised state-of-the-art. Chapter 5 forgoes transfer learning from ImageNet to instead focus on the problem of class generalization. An architecture for single-image 3D reconstruction is presented which improves generalization to previously unseen classes by leveraging an inputted class shape prior at inference time. In Chapter 6, we consider a domain where deep learning has yet to establish itself in timeseries analysis. A novel architecture is presented for deep metric learning on timeseries that achieves state-of-the-art on classification tasks given unsupervised pretraining. This work is still in progress, as we hope to create networks

which can provide the same generalizable power on timeseries that ImageNet-pretrained CNNs have provided to computer vision. Finally, I conclude with Chapter 7 where I reflect on the future growth of this work as well as the field in general.

As a clarifying note, the body chapters of this thesis are all adaptations of published papers or papers under review, with the exception of Chapter 6 which was adapted from a partial manuscript. Given the non-linear nature of these chapters (and correspondingly my research), I leave each chapter to serve as a stand-alone module with separate background, related work, and notation. While this results in some amount of informational redundancy and notational disagreement, I believe that it improves the overall reading experience. The supplementary material and appendices of the individual chapters have been consolidated into the chapters themselves where relevant.

CHAPTER 2

EXTENDING AND ANALYZING SELF-SUPERVISED LEARNING ACROSS DOMAINS

2.1 Introduction

A good visual representation is key to all visual recognition tasks. However, in current practice, one needs large labeled training sets to train such a representation. Unfortunately, such datasets can be hard to acquire in many domains, such as satellite imagery or the medical domain. This is often either because annotations require expertise and experts have limited time, or the images themselves are limited (as in medicine). To bring the benefits of visual recognition to these disparate domains, we need powerful representation learning techniques that do not require large labeled datasets.

A promising direction is to use self-supervised representation learning (SSRL), which has gained increasing interest over the last few years[60, 142, 243, 69, 100, 244]. However, past work has primarily evaluated these techniques on general category object recognition in internet imagery (e.g. ImageNet classification)[163]. There has been very little attention on how (and if) these techniques extend to other domains, be they fine-grained classification problems or datasets in biology and medicine. Paradoxically, these domains are often most in need of such techniques precisely because of the lack of labeled training data.

As such, a key question is whether conclusions from benchmarks on self-supervised learning [69, 100] which focused on internet imagery, carry over to

this broader universe of recognition problems. In particular, does one technique dominate, or are different pretext tasks useful for different types of domains (**Sec. 2.5.1**)? Are representations from an ImageNet classifier still the best we can do (**Sec. 2.5.1**)? Do these answers change when labels are limited (**Sec. 2.5.1**)? Are there problem domains where all proposed techniques currently fail (**Sec. 2.5.2**)?

A barrier to answering these questions is our limited understanding of self-supervised techniques themselves. We have seen their *empirical* success on ImageNet, but when they do succeed, what is it that drives their success (**Sec. 2.5.3**)? Furthermore, what does the space of learned representations look like, for instance in terms of the dimensionality (**Sec. 2.6.1**) or nearest neighbors (**Sec. 2.6.2**)?

In this work, we take the first steps towards answering these questions. We evaluate and analyze multiple self-supervised learning techniques (Rotation[60], Instance Discrimination[220] and Jigsaw[142]) on the broadest benchmark yet of 16 domains spanning internet, biological, satellite, and symbolic imagery. We find that Rotation has the best overall accuracy (reflective of rankings on ImageNet), but is outperformed by Instance Discrimination on biological domains (**Sec. 2.5.1**). When labels are scarce, pretext methods outperform ImageNet initialization and even full supervision on numerous tasks (**Sec. 2.5.1**). A prominent failure case for SSRL is fine-grained classification problems, due to important cues such as color being discarded during training (**Sec. 2.5.2**). Finally, when SSRL techniques do succeed, their reason for success varies: Rotation relies more on the semantic nature of the pretext task, compared to Jigsaw and Instance Discrimination (**Sec. 2.5.3**). Perhaps as a con-

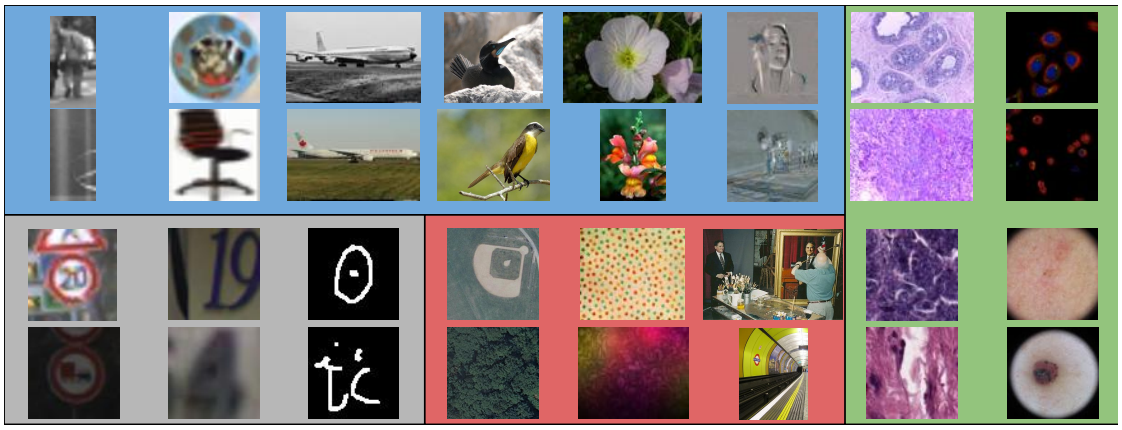

Figure 2.1: Samples from all datasets. Top rows: Daimler Pedestrians, CIFAR100, FGVC-Aircraft, CU Birds, VGG-Flowers, UCF101, BACH, Protein Atlas. Bottom rows: GTSRB, SVHN, Omniglot, UC Merced Land Use, Describable Textures, Indoor Scenes, Kather, ISIC. Color coding is by group: Internet Symbolic Scenes & Textures Biological

sequence, the representations of Rotation having comparatively higher implicit dimensionality (**Sec. 2.6.1**).

2.2 Datasets

We include 16 datasets in our experiments, significantly more than all prior work. Dataset samples are shown in Figure 2.1. We group these datasets into 4 categories: **Internet**, **Symbolic**, **Scenes & Textures**, and **Biological**. A summary is shown in Table 2.1. Some of the datasets in the first three groups are also in the Visual Domain Decathlon (VDD)[158], a multi-task learning benchmark.

Internet Object Recognition: This group consists of object recognition problems on internet imagery. We include both coarse-grained (CIFAR100, Daimler Pedestrians) and fine-grained (FGVC-Aircraft, CUB, VGG Flowers) object clas-

Table 2.1: Summary of the 16 datasets included in our experiments: encompassing fine-grain, symbolic, scene, textural, and biological domains. This is the first work exploring self-supervised representation learning on almost all of these tasks

Name	Type	Size (Train)	Coarse/Fine	Abbreviation
Daimler Pedestrians[138]	Road Object	20k	Coarse	PED
CIFAR100[103]	Internet Object	40k	Coarse	C100
FGVC-Aircraft[121]	Internet Object	3.3k	Fine	AIR
Caltech-UCSD Birds[215]	Internet Object	8.3k	Fine	CUB
VGG-Flowers[141]	Internet Object	1k	Fine	FLO
UCF101[175, 16]	Pseudo-Internet Action	9.3k	Coarse	UCF
German Traffic Signs [177]	Symbolic	21k	Coarse	GTS
Street View House Numbers[139]	Symbolic	59k	Coarse	SVHN
Omniglot[106]	Symbolic	19k	Fine	OMN
UC Merced Land Use[231]	Aerial Scene	1.5k	Coarse	MER
Describable Textures[37]	Texture	1.9k	Fine	DTD
Indoor Scene Recognition [154]	Natural Scene	11k	Coarse	SCE
ICIAR BACH[7]	Biological	240	Coarse	BACH
Kather[93]	Biological	3k	Coarse	KATH
Protein Atlas[145]	Biological	9k	Fine	PA
ISIC[38, 189]	Biological	17k	Coarse	ISIC

sification tasks. Finally, we include the "dynamic images" of UCF101, a dataset that possesses many of the same qualitative attributes of the group.

Symbolic: We include three well-known symbolic tasks: Omniglot, German Traffic Signs (GTSRB), and Street View House Numbers (SVHN). Though the classification problems might be deemed simple, these offer domains where classification is very different from natural internet imagery: texture is not generally a useful cue and classes follow strict explainable rules.

Scenes & Textures: These domains, UC Merced Land Use (satellite im-

agery), Describable Textures, and Indoor Scenes, all require holistic understandings, none having an overarching definition of object/symbol. Indoor Scenes does contain internet imagery as in our first group, but is not object-focused.

Biological: BACH and Kather consist of histological (microscopic tissue) images of breast and colon cancer respectively, with the classes being the condition/type of cancer. Protein Atlas is microscopy images of human cells, with the goal being classification of the cell part/structure shown. Finally, ISIC is a dermatology dataset consisting of photographs of different types of skin lesions.

Before evaluating self-supervision, we study the datasets themselves in terms of difficulty and similarity to ImageNet. To do so, we compare the accuracy of a network trained from scratch with that of a linear classifier operating on an ImageNet-pretrained network (Figure 2.2). The higher of the two numbers measures the difficulty, while their relationship quantifies the similarity to ImageNet.

We find that small datasets in the Internet domain tend to be the hardest, while large Symbolic datasets are the simplest. The symbolic tasks also have the largest gap between supervision and feature extraction, suggesting that these are the farthest from ImageNet. Overall, the ImageNet feature extractor performance is strongly linearly correlated to that of the fully supervised model ($p = 0.004$). This is expected for the Internet domain, but the similar utility of the feature extractor for the Biological domains is surprising. Dataset size also plays a role, with the pretrained feature extractor working well for smaller datasets.

For fine-grained classification (AIR, CUB, FLO), the supervised models per-

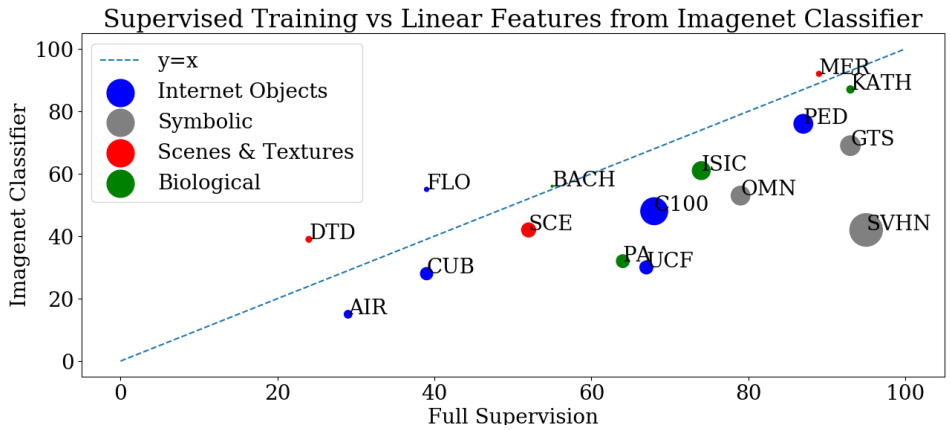

Figure 2.2: Test accuracy of a fully supervised network vs. a linear classifier on top of an ImageNet-classification frozen feature extractor. Marker area indicates dataset size

form comparably, while the ImageNet classifier's performance varies widely. In addition to the small size of VGG-Flowers, this case is also partly explainable by how these datasets overlap with ImageNet. Almost half of the classes in ImageNet are animals or plants, including flowers, making pretraining especially useful.

2.3 Methods

2.3.1 Self-Supervised Learning Techniques

In this paper we look at three popular methods, Rotation, Jigsaw, and Instance Discrimination. We also look at the classical technique of Autoencoders as a baseline for the large variety of autoencoder-based pretexts[244, 243, 150]. We briefly describe each method below, please view the cited works for detailed information.

Learning by Rotation: A network is trained to classify the angle of a rotated image among the four choices of 0, 90, 180, or 270 degrees[60].

Learning by Solving Jigsaw Puzzles: The image is separated into a 3x3 grid of patches, which are then permuted and fed through a siamese network which must identify the original layout[142, 21]. We use 2000 training permutations in our experiments, finding this offered superior performance to 100 or 10,000 under our hyperparameters.

Instance Discrimination: Instance Discrimination (ID) maps images to features on the unit sphere with each image being considered as a separate class under a non-parametric softmax classifier[220].

Autoencoders: Autoencoders were one of the earliest methods of self-supervised learning[244, 198, 150, 14, 80]. An autoencoder learns an encoder-decoder pair of networks that reconstruct the input image.

2.3.2 Architectures

ResNet26

See Figure 3 and Section 3.2 of [158] for the original description. Features maps of size $256 \times 8 \times 8$ are extracted before the final pooling layer and average pooled to 256, 4096 ($256 \times 4 \times 4$), or 9216 ($256 \times 6 \times 6$), with 256 being the default. A linear classifier is trained on these features.

Layer	Output Shape
Input	256
ConvTranspose2d-1	[-1, 512, 4, 4]
BatchNorm2d-2	[-1, 512, 4, 4]
ReLU-3	[-1, 512, 4, 4]
ConvTranspose2d-4	[-1, 256, 8, 8]
BatchNorm2d-5	[-1, 256, 8, 8]
ReLU-6	[-1, 256, 8, 8]
ConvTranspose2d-7	[-1, 128, 16, 16]
BatchNorm2d-8	[-1, 128, 16, 16]
ReLU-9	[-1, 128, 16, 16]
ConvTranspose2d-10	[-1, 64, 32, 32]
BatchNorm2d-11	[-1, 64, 32, 32]
ReLU-12	[-1, 64, 32, 32]
ConvTranspose2d-13	[-1, 3, 64, 64]
Tanh-14	[-1, 3, 64, 64]

Table 2.2: Generator architecture. First convolution has stride of 1 and no padding, all subequent convolutions have stride of 2 with padding 1. All kernels have size 4.

Autoencoder Generator

We resize inputs to 64×64 and use a ResNet26, as in Rebuffi et al. [158, 159]. The lower resolution eases computational burden as well as comparison and future adaptation to the VDD. For Autoencoding, a simple convolutional decoder is used. The architecture is laid out in Table 2.2.

2.3.3 Training & Evaluation

All networks are trained using stochastic gradient descent for 120 epochs with an initial learning rate of 0.1 decayed by a factor of 10 at 80 and 100 epochs, with momentum of 0.9. One addition to our training process was that of "Ear-

lier Stopping" for the Rotation and Jigsaw pretext tasks. We found that even with traditional early stopping, validation accuracy could oscillate as the pretext overfit to the training data (especially in the Scenes & Textures or Biological cases), potentially resulting in a poor model as the final result. We stabilized this behavior by halting training when the training accuracy improved to 98%.

2.4 Related Work

There are three pertinent recent surveys of self-supervision. The first is by Kolesnikov et al. who evaluate several methods on ImageNet and Places205 classification across network architectures[100]. We focus on many domains, an order of magnitude more than their work. The second relevant survey is by Goyal et al., who introduce a benchmark suite of tasks on which to test models as feature extractors. While they scale on a variety of downstream tasks, the pretraining datasets are all internet imagery, either ImageNet or YFCC variants[69, 185]. Our work includes a much wider variety of both pretraining and downstream datasets. VTAB tests *pretrained feature extractors* on a variety of datasets, performing self-supervised learning only on ImageNet[240]. Finally, a concurrent paper evaluates these self-supervised techniques as an auxilliary loss for few-shot learning[180].

One trend of inquiry concerns classifiers that perform well on multiple domains while sharing most of the parameters across datasets[158, 159, 210]. The pre-eminent examples of this are by Rebuffi et al. [158, 159] who present approaches on the VDD across 10 different domains. We use these datasets (and more) in our training, but evaluate *self-supervised approaches* in *single-domain* set-

tings.

There has also been prior work using problem/domain-specific SSRL methods, such as [87, 117, 28, 56] in the biological and medical fields or [164] for aerial imagery. Many of these approaches use variations of autoencoding as the pretext task; we include autoencoding in our evaluation. In contrast to these, our focus is on the cross-dataset applicability of these pretexts.

Other SSRL methods include generative models[51, 54, 64], colorization [107, 243], video-based techniques[146, 45, 149, 134, 67, 201], or generic techiques[23, 17, 76, 133, 29]. It is very possible that a subset of these methods could offer improved performance on some of the domains we work with, however in this work we focus on the popular fundamental methods of Rotation, Jigsaw, and Instance Discrimination as well as Autoencoding as a representative set. Doersch and Zisserman use multiple pretexts simultaneously to improve representation quality [50]. This approach could be complementary to the work featured here. Semi-supervised approaches such as S4L[239] are relevant to Section 2.5.1.

2.5 Downstream task Performance Analysis

2.5.1 Downstream task accuracy

A summary of downstream testing accuracies is shown in Figure 2.3. The obvious question to ask in this investigation is "which pretext is best"? On ImageNet, per the respective original works, the ordering (best to worst) was Rotation, Instance Discrimination, Jigsaw, Autoencoding. We see that this ranking is

not universal: while Rotation is the best pretext on the first three groups, it lags behind even random initialization on the Biological domains. Furthermore, the relative rankings of ID and Jigsaw vary between groups. We investigate what powers the performance of each of these methods in Section 2.5.3.

Limited Label Training

Self-supervised learning has made an impact in the field of *semi*-supervised learning, where only a subset of the dataset given is labeled, as in the work of Zhai et al.[239]. In that work on ImageNet, a self-supervised feature extractor followed by a supervised linear head falls well short of the purely supervised baseline (40% accuracy compared to 80% when 10% of labels are available). We find that this conclusion does not hold on our domains when 10% of labels are used, with a pretext + linear setup outperforming the fully supervised models on some datasets/groups (Figure 2.4).[1]

On the Internet and Scenes & Textures groups, Rotation matches/outperforms full supervision. On ISIC, both Instance Discrimination and Jigsaw match Supervision. Interestingly, Autoencoding performs well in the Biological domains. Given the difficulty of expert annotation in the medical field, this is a valuable finding to encourage Autoencoder-based methods in these label-scarce domains, vindicating choices in past work[87, 117, 28, 56]. The unlabeled 90% of the data being available for SSRL methods is critical: when only the labeled subset is used for pretraining the performance drops an average of approximately 10%.

[1]Note that [239] performs extensive hyperparameter tuning for the supervised baseline.

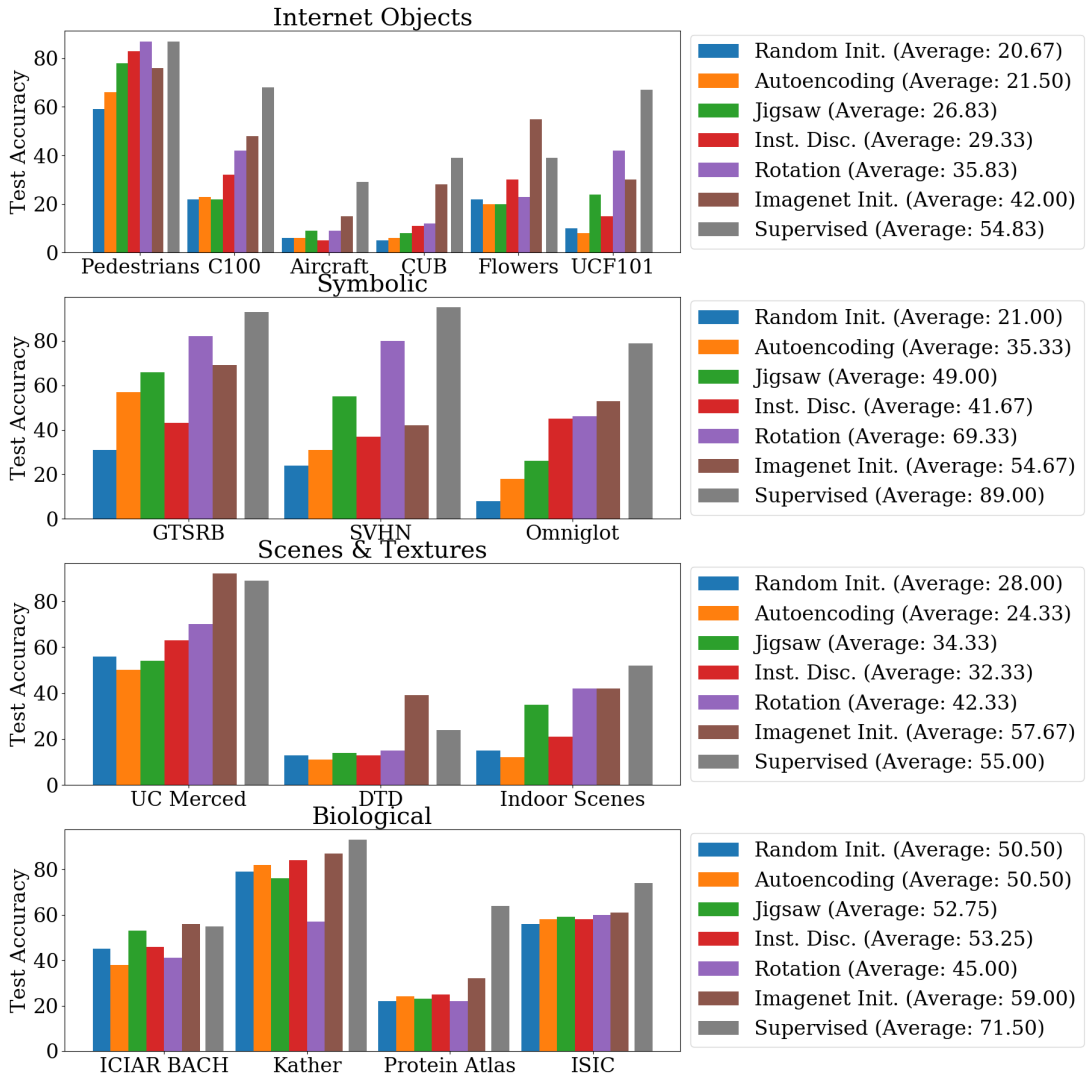

Figure 2.3: Downstream classification accuracies for each pretext, as well as a randomly initialized feature extractor, a frozen ImageNet classifier, and fully supervised training. Rotation achieves the highest accuracies on the first three groups, but fails on the Biological domain, where Instance Discrimination performs best. The relative rankings of Jigsaw and ID vary between groups and are practically equal in overall average

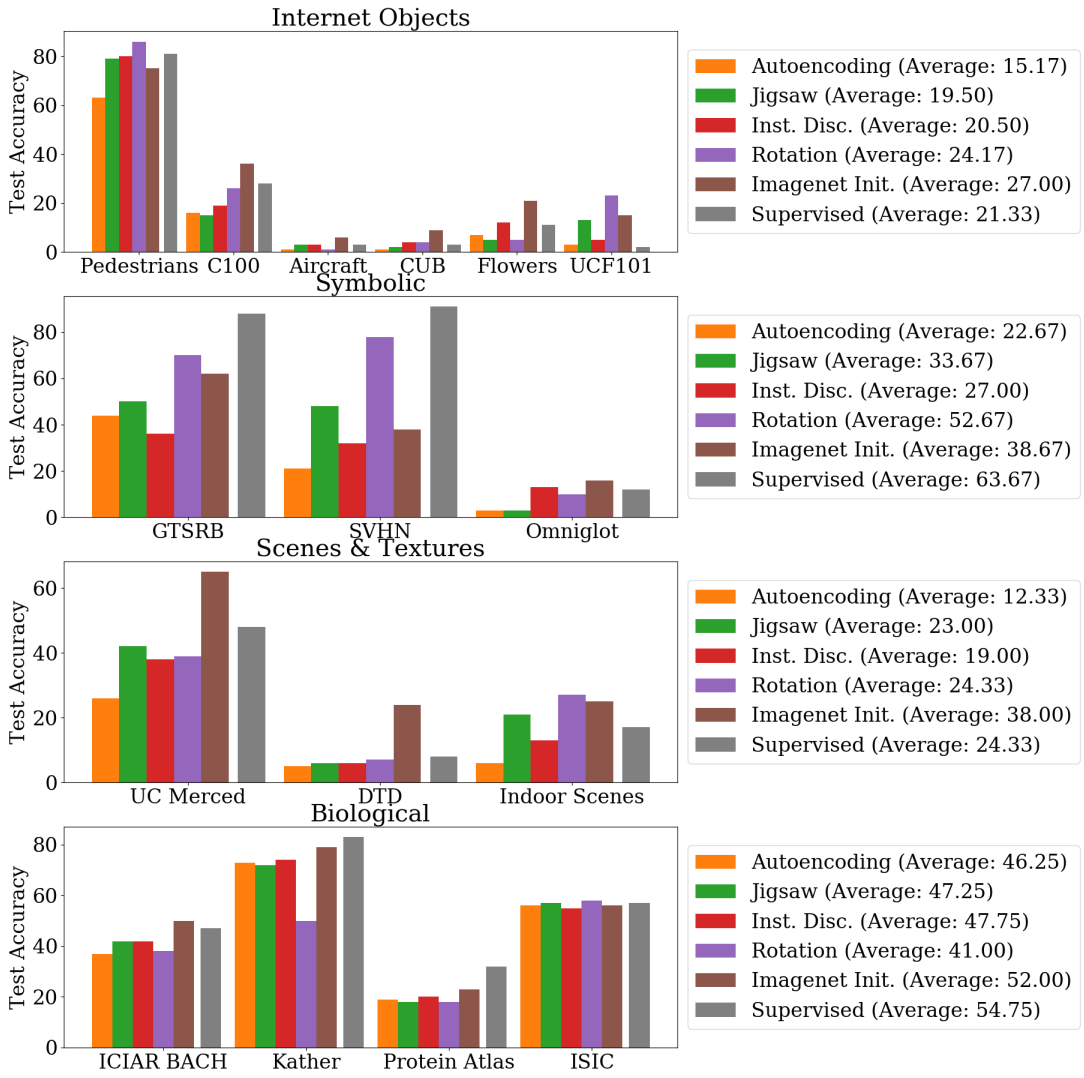

Figure 2.4: Pretext feature extractors trained on the entire dataset, then a linear head trained on 10% of the labels. The fully supervised method simply trains on 10% of the dataset. We observe Rotation matching/outperforming Supervision for Internet and Scenes & Textures imagery. Autoencoding performs very well on the Biological domains in the semi-supervised setting, a novel result that indicates the potential for future development (given that this Autoencoder implementation is the simplest possible)

2.5.2 Inspecting Failure Modes

We call specific attention to the problems where self-supervised techniques do not achieve even *half* of the supervised accuracy (Aircraft, CUB, Textures and Protein Atlas). These seem to involve two kinds of problems.

Textures and Protein Atlas:

In both these datasets, the entities being classified are not objects of recognizable shape, which is true of most *object recognition* datasets where the self-supervised techniques were developed. The images also do not have a canonical orientation, unlike in internet imagery where gravity and photographers' biases provide such an orientation. We hypothesize that the lack of orientability hobbles Rotation, and the textural nature of both problems results in there being little to distinguish a patch in the Jigsaw pretext from a complete image, meaning Jigsaw can do little besides cue off of low-level properties (such as chromatic aberrations [142]) As such, modulo low-level dataset biases, the Rotation and Jigsaw pretext tasks *cannot even be solved in these domains*.

Fine-Grained Classification on Aircraft and CUB:

In these datasets, the Rotation and Jigsaw pretexts are solvable, involving objects captured in a canonical orientation. Even so, neither pretext learns a good representation. While the *domains* are favorable to the pretexts, the *tasks* are not: both tasks involve fine-grained distinctions in color or texture, and subtle differences in shape. One hypothesis is that modeling these subtle differences is not necessary for solving the pretext task, causing the learnt representation to

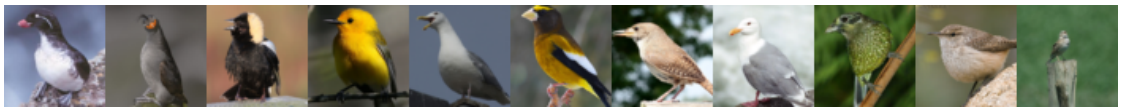

Figure 2.5: The 10 nearest neighbors (in order) of the far-left image in the CUB validation set measure by the unpooled feature space of the Rotation pretext model. We see that coloring plays almost no role in determining neighbors, but pose is a very strong signal

be *invariant* to these vital distinctions. Indeed, when we look at nearest neighbors in CUB for our Rotation model, we see birds of completely different colors but in the same pose, suggesting that the representation has captured pose, but ignored color (Figure 2.5). We quantitatively evaluate this hypothesis further in Section 2.5.3.

Note that we have not discussed the failures of Instance Discrimination in both domains. Instance Discrimination must learn to distinguish between individual images, which should be solvable in any domain as long as the images are distinct. As such, its failure and general performance is much more unpredictable and less interpretable, as we elaborate on in the next section.

2.5.3 Reasons for Success

The flip side of why they fail is why they succeed. In prior work on the domain of internet imagery, this has mostly been answered by intuition. Rotation and Jigsaw were engineered as tasks that ostensibly require semantic understanding to solve. Instance Discrimination's success is similarly intuitive: spreading points in a constrained space will naturally cluster similar images. Given the failures above, neither of these intuitions endure without modification in the

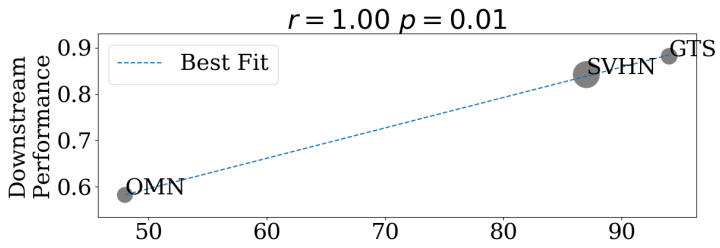

Figure 2.6: Rotation pretext testing accuracy vs. unnormalized downstream performance. Almost perfect correlation is evident

varied domains of our work, and a more nuanced reasoning is necessary.

Semantic understanding: As discussed above, for some domains such as Aircraft or CUB, the pretexts do not produce a good semantic representation perhaps because they do not require a semantic understanding to solve. As an additional example, Jigsaw classifies permutations on Omniglot with 77% accuracy, but performs poorly on the classification task: line-matching is not understanding. Another example is Kather, where Rotation picks up on some hidden cue to fully solve the pretext problem without attaining any semantic knowledge. On the other side of the spectrum, we observe on the Symbolic domains in Figure 2.6 that Rotation *is* implicitly performing semantic classification, making it a near-optimal pretext task. *When* is semantic understanding a prerequisite for a pretext task, and how can we test this?

We propose that semantic understanding is a prerequisite for a pretext *when the solution method is class-dependent*: in this case, the network must necessarily implicitly classify the image to perform the pretext well. For example, in the fine-grained classification of airplanes and birds, determining the orientation of the object is class-*in*dependent, so a network trained on the Rotation pretext does not need to learn features indicative of class. One way of testing if the

pretext solution is class independent is to see if a network trained to solve the pretext on one set of classes solves the pretext on *unseen* classes. We call this test *pretext generalization* (Figure 2.7): we train and validate pretexts on only half of the available classes, and then evaluate the pretext on the entire test set. We predict that worse pretext generalization should imply better downstream performance.

Our prediction is correct for Rotation, affirming our hypothesis, but Jigsaw models seem to show high pretext generalization in almost all cases. This lack of correlation might be because Jigsaw is relying on low-level cues that do generalize *in addition to* the semantic information. We contrast these findings with a concurrent paper[180] which speculates that Rotation is not as useful for few-shot learning on FGVC-Aircraft or VGG-Flowers due to the relative difficulty of the pretext task.

For Instance Discrimination, counterintuitively, downstream accuracy is *not* correlated with pretext loss (Figure 2.7B). Our proposal is thus not the complete picture.

Linear separability: The above discussion assumes that the relationship between the pretext task and the downstream task is the key. But what if the downstream task is immaterial? Pretext tasks might be succeeding by simply enabling *all* tasks, by creating a feature space where *many* labelings of the data points are expressible as linear classifiers. To test this hypothesis, we retrain linear classifiers with randomly shuffled training labels and compare this (training) accuracy to the training accuracy with correct labels. We find that these quantities are more correlated for Jigsaw than Instance Discrimination or Ro-

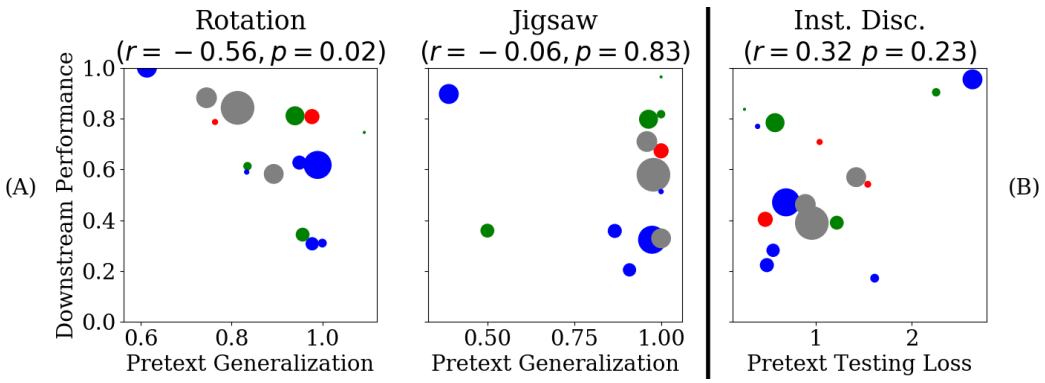

Figure 2.7: **(A)** Generalization of the pretexts to novel classes is plotted on the x-axis, calculated as $\frac{PretextAcc_{Test}}{PretextAcc_{Val0.5}}$, where Train/Val0.5 contain only half the classes of their usual counterparts. On the y-axis is downstream accuracy normalized by Supervision accuracy *for the regularly trained model*, e.g. $\frac{ClassAcc_{Rot}}{ClassAcc_{Sup}}$. We see a strong correlation for Rotation, but none for Jigsaw. **(B)** Instance Discrimination pretext loss vs. downstream classification accuracy, no significant correlations. Marker area indicates dataset size and color the domain grouping.

Table 2.3: Pearson correlations and p-values for training accuracy with random labels vs. normal. Jigsaw and ID have significant correlations, while Rotation trails substantially

	Jigsaw		Inst. Disc.		Rotation.	
(r, p)	0.59	0.02	0.46	0.07	0.45	0.08

tation (Table 2.3), per task decreased are shown in Figure 2.8. This means that Jigsaw succeeds more by learning generic feature descriptors of images rather than by capturing semantics specific to the downstream task, while this is less true for Rotation or Instance Discrimination. Note that here we are talking of the training accuracy, or the empirical risk; this experiment does not reveal how or why pretext methods generalize to data not in the training set.

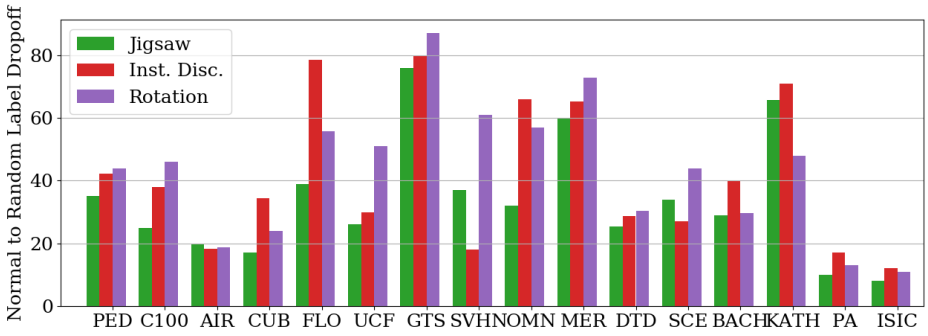

Figure 2.8: Difference between normal and random label training accuracy. Note that even though Instance Discrimination had lower average *validation* accuracy than Rotation across all datasets, they actually had similar (normal label) average *training* accuracy at 56% (Jigsaw had 47%).

2.6 Feature Space Exploration

The discussion above suggests the virtue of analyzing the learnt representations independent of the downstream task. Below, we look at the *intrinsic dimensionality* of the learned representations, and the resulting notions of similarity.

2.6.1 Implicit Dimensionality of the Representations

The dimensionality of the representation is largely regarded as a hyperparameter to be tuned. Kolesnikov et al. show that a larger representation is always more useful when operating on large datasets, but this comes with a tradeoff of memory/storage[100]. What has not been studied is the *implicit* dimensionality of the representations, such as via Principal Component Analyis (PCA). Intuitively, one would expect the 4-way Rotation task to produce compact representations, with Instance Discrimination using all available dimensions to spread

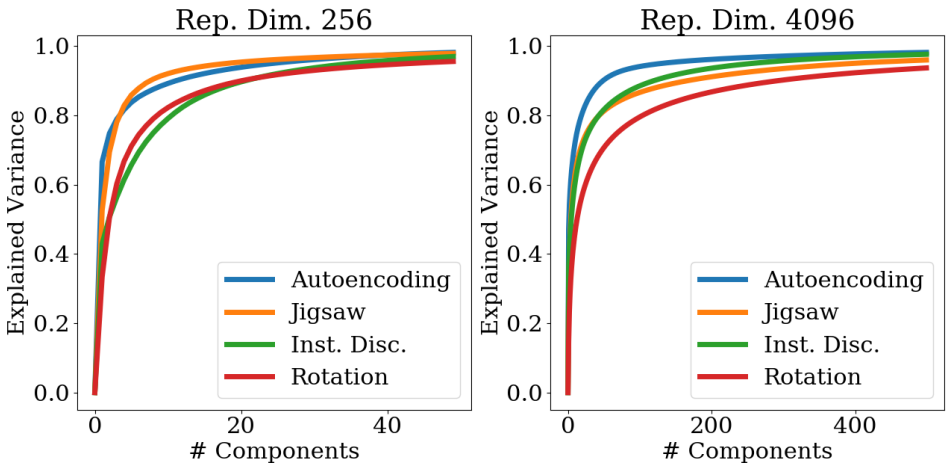

Figure 2.9: Plot of the average fraction of explained variance vs. number of principal components (validation sets); a higher number means greater explanation, and thus relatively *lower* implicit dimension. Bach is omitted due to its extremely small size

out the points as the loss demands. We find that this is *not* the case, via performing PCA on the representations and summing the explained variance of the first n values averaged across all datasets (Figure 2.9). We also find that the implicit dimensionality induced by each pretext varies considerably across domains (Figure 2.10).

Surprisingly, Instance Discrimination and Rotation have extremely similar implicit dimensionality in the 256-dimensional case. Also note that Instance Discrimination clearly is not fully utilizing the available latent space, as the first 40 components explain over 95% of the variance. In the higher-dimensional example, Rotation has by far the largest implicit dimensionality.

We next investigate the effect of dataset size on implicit dimensionality. Intuition says that a larger dataset would demand a more expressive representation on every task, and we see in Figure 2.10 that this indeed holds true for *all* tasks except for Jigsaw. We hypothesize the lack of dimensionality increase for Jigsaw

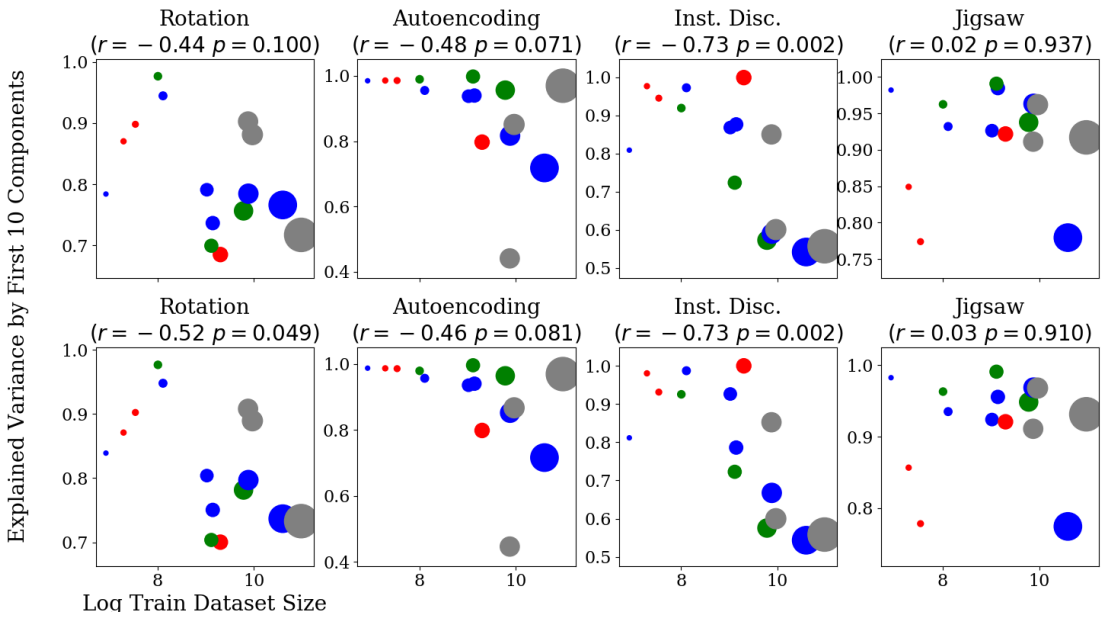

Figure 2.10: The fraction of variance explained by the first 10 components vs. the log of the training set size. Top row is training PCA, bottom row is validation PCA. Moderate to strong correlations exists for all pretext tasks besides Jigsaw. Marker area indicates dataset size and the colors are domain groupings.

is because it exploits relatively low-level attributes instead of semantic knowledge.

2.6.2 Nearest Neighbors

As seen in Figure 2.5, the nearest neighbors in feature space can yield great insight into the inner workings of our models. We repeat this experiment for all of our main methods on three datasets (CUB, Omniglot, Kather) in Figure 2.11.

CUB: The strong pose-capturing of Rotation is again observed. Jigsaw exhibits similar phenomenon slightly more weakly, and also favors similarly cluttered edge-heavy backgrounds as in the base image. Instance Discrimination

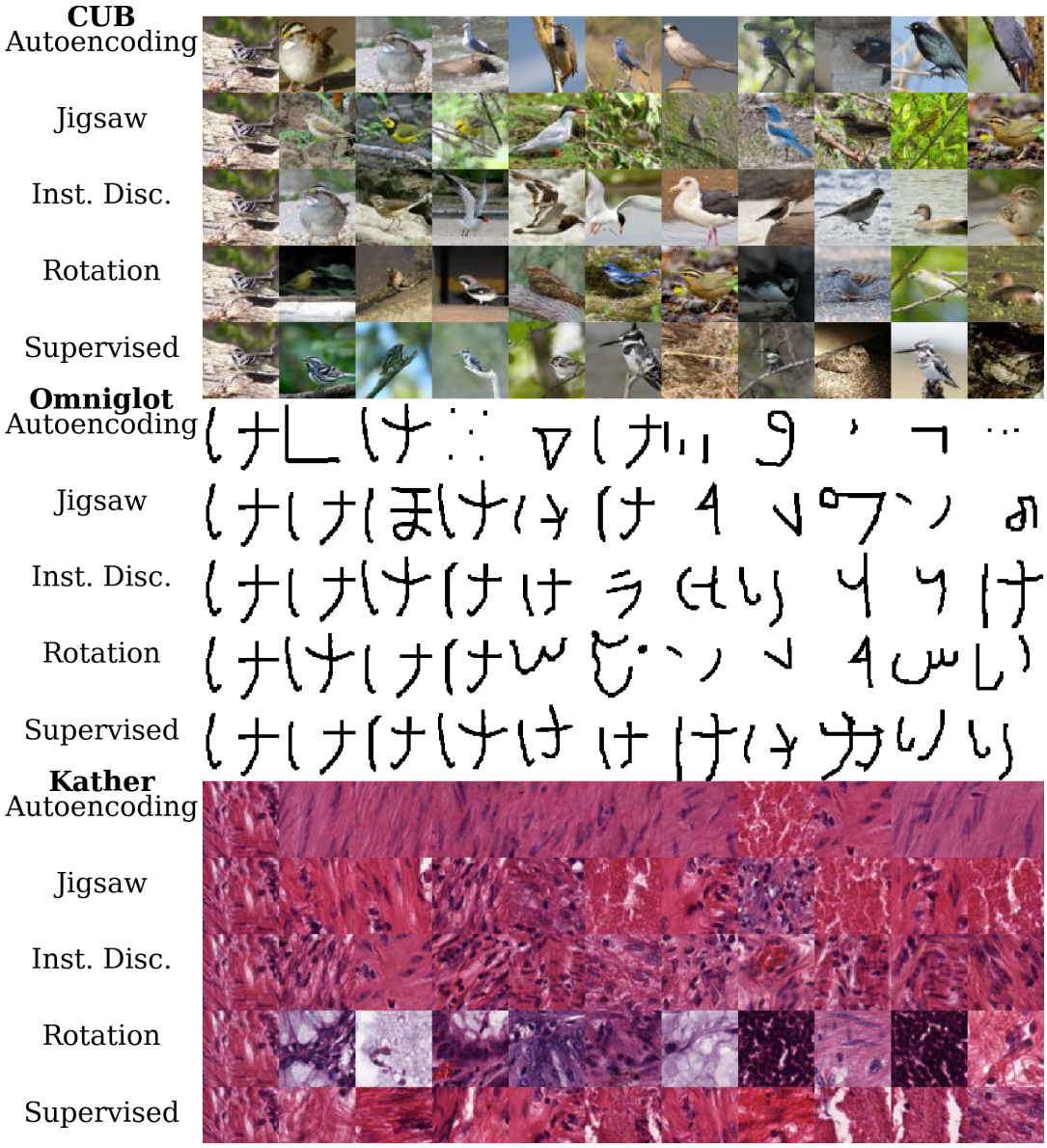

Figure 2.11: Each block of images comes from a single dataset, with each row corresponding to the pretext method. The farthest left image is the base image from which the distance in feature space is calculated to all the others in the validation set. Left-to-right is increasing distance in feature space among the 10 nearest neighbors

appears to have reasonably matched the species in the first and tenth neighbor, but in between has many unrelated birds.

Omniglot: The poor performance of Autoencoding is reflected in its matching ability. Rotation exhibits strong retrieval, with Jigsaw offering similar but weaker results. Instance Discrimination is a similar case to CUB, where a discernable trend is hard to spot, but several correct characters are matched.

Kather: No method improved much on random initialization (79% accuracy). Reflectively, Autoencoding, Jigsaw, and ID all simply match reasonably visually similar results. Rotation, however, failed catastrophically (<60% accuracy), despite high pretext accuracy (95% testing). We hypothesized that the pretext was exploiting a spurious cue, which is affirmed by radically varying neighbors.

2.7 Conclusion

In this work, we have for the first time explored the notion of self-supervised learning in small-scale images in novel domains, identifying three domains where all current self-supervised methods have need of development: fine-grained, textural, and biological domains. In addition, we have revealed intriguing properties of the pretexts and the corresponding learnt representations, whose impact deserves further study. We hope that the release of our codes, models, and formatted dataset splits will help aid progress on all of these fronts.

CHAPTER 3

LEARNING RICH NEAREST NEIGHBOR REPRESENTATIONS FROM SELF-SUPERVISED ENSEMBLES

3.1 Introduction

The widespread application of pretrained convolutional neural networks in computer vision is one of the most important tools in the field. State-of-the-art on many benchmarks ranging from classification, to object detection, to pose estimation has been set using a pretrained model, such as an ImageNet classifier, as a network initialization [101, 76, 29, 72, 99]. Transfer learning is an entire field focused on studying and utilizing this phenomenon. While supervised ImageNet classifiers were the dominant feature extractors of choice for many years, recently self-supervised models have begun to take their place. Methods such as MoCo(v2), SimCLR(v2), SimSiam, PIRL, Swav, BYOL, and Barlow Twins all claim transferability competitive with or *superior to* that of ImageNet classifiers [76, 31, 29, 30, 32, 133, 25, 72, 236]. As such, the question of what initialization to use has arisen; benchmark studies have sought to compare methods under dozens of different settings [70, 240]. Even when a decision has been made to use a particular feature extractor, the utility and knowledge of other options is then left unutilized.

To address this concern, we consider ensembling, a common practice in the supervised setting [49, 79]. Ensembling models involves combining the predictions obtained by multiple different models in some way, typically by averaging or a similar operation. In the supervised setting, such outputs are aligned and such an operation easily captures the combined knowledge of the models. In the

self-supervised setting, however, such alignment is not guaranteed, particularly when dealing with independently trained contrastive learners which many pre-trained models of choice are. Averaging the features still is useful, and obtains reasonably strong image representations (Section 3.4), but we show that it is possible to build an ensembling strategy that yields richer, stronger representations than the mean feature. We do so without training any new CNN components, allowing for the same frozen backbone to be used across applications.

How to approach ensembling in the self-supervised setting? We contend that *the goal of a model ensemble is to capture the useful information provided by the different models in a single representation*. We consider the "capture" of information from a recoverability perspective: if every network's features can be recovered by some fixed operation on a representation vector, for all data samples, then such representations are useful. While concatenation of features can trivially achieve this object, we show that such an operation is in fact suboptimal in terms of the behavior of the derived feature space. We instead propose to *directly* learn via gradient descent a set of representations that contain all of the information necessary to derive the ensemble features. Our architecture is shown in Figure 3.1, with example pseudocode in Algorithm 1. Specifically, we show that extracting features from an ensemble of self-supervised models using this technique improves the nearest neighbor (NN) performance when evaluated on downstream supervised tasks.

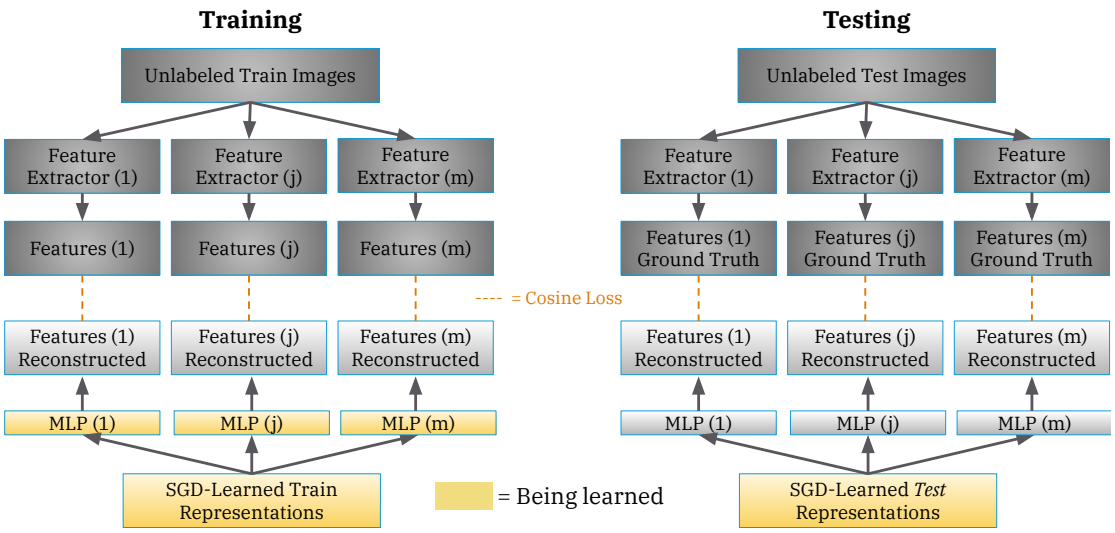

Figure 3.1: A schematic of our method. We wish to ensemble M feature extractors over a training dataset of n images, M MLPs are initialized as well as representations for each of the n images. The training objective is for all of the features of the M models to be recoverable by feeding the learned representations through the respective MLP. The MLPs and learned representations are simultaneously optimized via gradient descent, using a cosine loss. At inference time, the MLPs are frozen, and solely the image representation is optimized.

3.2 Related Work

Supervised ensembling: It is a ubiquitous technique in machine learning [49]. In addition to the large number of online contests won through such approaches [5], ensembling has also been demonstrated to achieve state-of-the-art performance on standard computer vision benchmarks [82]. Most approaches employ Bayesian ensembling, where the predictions of separate networks are averaged [219, 79, 49]. While this relied on the alignment of objectives between the networks, we show that such an averaging on the intermediate features does indeed generate representations superior to that of individual models. Our

Algorithm 1: PyTorch-like pseudocode of our method

```
# Training Phase
# Initialize representations to average feature
train_feature_list = [net(images) for net in ensemble]
avg_feat = average(train_feature_list) # shape = (n_points x feature_dimension)
learned_train_reps = Parameter(avg_feat.detach()) # initialize params with avg_feat
mlps = [MLP() for net in ensemble] # 1 net per feature extractor
opt = SGD(mlps.parameters() + learned_train_reps) # optimize both mapping MLPs and input
     representations
for images_idx, images in trainloader:
   ensemble_feats = [net(images) for net in ensemble] # Get ensemble features
   outputs = [mlp(learned_train_reps[images_idx] for mlp in mlps] # Map learned representations
        through different MLPs
   loss = cosine_loss(ensemble_feats, outputs)
   loss.backward()
# Inference Phase
test_feature_list = [net(images) for net in ensemble]
learned_test_reps = average(test_feature_list)
opt = SGD(learned_test_reps) # Freeze MLPs at inference time
for images_idx, images in testloader:
   # Same as training loop
```

MLP(): a multi-layer perceptron model
Parameter(t): Pytorch function that takes the argument array t and initializes trainable parameters with that value

method differs from this literature, however, in the learning done on top of the ensemble as well as the no-label setting of our representation learning.

Knowledge distillation [79]: This is a related vein of work is that of where the knowledge of an ensemble is distilled into a single neural network, or a single model is distilled into another network. This second version, known as self-distillation [241], is highly relevant to our single-model setting, as we are able to obtain improvements while operating purely in the *feature* space of a single model. Our goal is not to discard the ensembled models as it is in knowledge distillation, but our method bears similarities to that of Hydra [124], where a network is trained to output representations capable of recovering the ensem-

bled outputs. We note that our resulting accuracies consistently surpass average ensembling, a baseline that Hydra considers as an upper-bound to their method (but the differing settings do not lend themselves to apples-to-apples comparisons).

[226] propose a self-supervised knowledge distillation algorithm that uses both supervised and self-supervised loss to train a teacher network, and then distill this combined knowledge to a student network. They show that this combination improves the student performance compared to traditional KD. In contrast to their work, our goal is not to learn a separate student network and we do not assume access to labeled data, but instead, our main goal is to extract rich nearest neighbor representations from an ensemble of self-supervised models.

***k*-Nearest neighbor (*k*-NN) classifiers**: *k*-NN is a non-parametric classifier which has been shown to be a consistent approximator ([48]), i.e., asymptotically its empirical risk goes to zero as $k \to \infty$ and $k/N \to 0$, where N is the number of training samples. While these theoretical conditions may not be true in practice, a main advantage of using *k*-NN is that it is parameter-free outside of the choice of k (which typically does not change qualitative rankings) and thus is a consistent and easily replicable measurement of representation quality. Additionally, *k*-NN makes the decision process interpretable [147, 42], which is important in various applications [129, 197, 61]. For these reasons, our paper focuses on extracting rich features from self-supervised ensembles that are condusive to *k*-NN classification.

On a related note, SLADE [53] leverages unlabeled data to improve distance metric learning, which is essentially the setting of our evaluation framework.

While the goal is similar, SLADE uses supervised training to initialize learning and generate pseudolabels for the unlabeled points, our method assumes *zero* label access. Additionally, SLADE is concerned with learning a new network from scratch, as opposed to an ensembling framework.

Pretrained Models: Our method relies on the efficacy of pretrained self-supervised models. Specific methods we employ are SimCLR, SwAV, Barlow Twins, RotNet, PIRL, as well as traditional label supervision. In addition to the above works: [70, 240, 99] benchmark and demonstrate the generalization efficacy of pretrained models in various settings.

Gradient descent at inference time: One atypical facet to our method is the use of gradient descent at inference time to directly learn new representations. While this approach is quite uncommon, we are not the first to leverage backpropagation at inference-time. [234] uses backpropagation at inference time to learn representations from images using a generative "auto-decoder" framework and [148, 170] employ similar approaches to learn implicit representations of shapes. [182] considers new samples as one-image self-supervised learning problems, and perform a brief optimization of a self-supervised learning objective on a new image (modifying the feature extractor's parameters) before performing inference, and [168] train a CNN at test-time for the purpose of super-resolution.

3.3 Methods

We present a method for directly learning rich ensembled unsupervised representations of images through gradient descent. Consider a training collection

of images $X = \{x_i\}_{i=1}^n$ and an ensemble of convolutional neural networks feature extractors $\Theta = \{\theta_j\}_{j=1}^m$. In this work the θ_j have previously been trained in a self-supervised manner on ImageNet classification and are ResNet-50s [163, 77]. Denote the L2-normalized features obtained by removing the linear/MLP heads of these networks and extracting intermediate features post-pooling (and ReLU) as $Z = \{\{z_i^{(j)}\}_{i=1}^n\}_{j=1}^m$, here $z_i^{(j)}$ denotes the intermediate feature corresponding to $\theta_j(x_i)$.

We initialize a memory bank of representations of X, with one entry for each x_i these entries have the same feature dimensionality as the z_i^j. This memory bank is analagous to the type use in early contrastive learning works such as [220]. We denote this memory bank $\Psi = \{\psi_k\}_{k=1}^n$. Each ψ_k is initialized to the L2-normalized average representation of the ensemble $\psi_k = \frac{\sum_{j=1}^m z_k^j}{\|\sum_{j=1}^m z_k^j\|}$, note that the sum operation is equivalent to averaging due to the normalization being performed.

To map the memory bank to the ensembled features, we employ a set of multi-layer perceptrons (MLPs), $\Phi = \{\phi_\ell\}_{\ell=1}^m$, each corresponding to a feature extractor θ_j. Unless noted otherwise in our experiments, these ϕ_ℓ are 2 layers, both of output dimension the same as their input (2048 for ResNet50 features). ReLU activations are used after *both* layers, the first as a traditional activation function, and the second to align the network in mapping to the post-relu set Z.

During training, a batch of images $\{x_i\}_{i\in I}$ are sampled with indices $I \subset \{1...n\}$. The corresponding ensemble features, $Z_I = \{\{z_i^{(j)}\}_{i\in I}\}_{j=1}^m$, are retrieved as are the memory bank representations $\Psi_I = \{\psi_k\}_{k\in I}$. Note that no image augmentations are included in our framework, meaning that the $z_i^{(j)}$ are typically cached to lessen computational complexity. Each banked representation is then fed through *each* of the m MLPs, Φ, resulting in a set of mapped representations

$\Phi(\Psi_I) = \{\phi_\ell(\psi_i)\}_{\ell \in \{1 \ldots m\}, i \in I}$. The goal of the training is to maximize the alignment of these mapped features $\Phi(\Psi_I)$ with the original ensemble features Z_I. This is done by training both the networks Φ and the representations Ψ using a cosine loss between $\Phi(\Psi_I)$ and Z_I, gradients are computed for both the MLPs and representations for each batch.

Once training is completed, the ϕ_ℓ are frozen. During inference, when a new image x' is given, the above process is repeated with the frozen MLPs. Concretely, the ensemble features $\phi_\ell(x')$ are computed and averaged to initialize a new representation ψ'. ψ' is then optimized via gradient descent to maximize the cosine similarity of each $\phi_\ell(\psi')$ with $\theta_\ell(x')$, ψ' then serves as our representation of x'.

The described method results in representations *superior to either the average or concatenated feature in terms of nearest-neighbor accuracy*. We highlight several exciting aspects of our method:

- The learning of a representation at test time via gradient descent is an uncommon approach. Existing methods do exist, such as auto-decoders in [234] or implicit 3D representation literature [148, 170]. There are also techniques, such as [182] for generalization, which use gradients to shape the *parameters* prior to inferenc.
- The vast majority of ensembling literature focuses on the *supervised* setting, where the training objectives of the ensembled networks are identical and thus aligned. Very little work has been performed on improving a group of self-supervised features with outside auxiliary signal. Hydra [124] considers a similar setting, but with a focus on knowledge distillation of the ensemble into a single network.

- Our method is extremely adaptable to different settings. Networks trained on multiple objectives, with different head architectures, can be usefully ensembled as demonstrated in Sec. 3.4. This could trivially be extended to using networks of different architectures as well (e.g. VGG + ResNet + AlexNet) [169, 77, 104]. The flexibility of our approach additionally extends to the input data. While we use networks pretrained on ImageNet, the ensembling provides benefit on *both ImageNet as well in the self-supervised transfer learning setting*. This transfer can either be performed by training new Φ on the target dataset or by training Φ on ImageNet and then using the frozen MLPs to learn representations on the target dataset.
- Our method as presented requires *only a single forward pass of each image through each ensembled model* as we do not use data augmentation. This allows caching of CNN features and fast training.

3.3.1 Technical Details

Models The MLPs are as previously described. The pretrained ensemble Θ consists convolutional neural networks (all ResNet50s with features extracted between the stem and head of the network) that have had pretraining on ImageNet. The method used in said pretraining varies and is a subject of study in this work. Methods considered include SimCLR(v2), SwAV, Barlow Twins, PIRL, Learning by Rotation (RotNet, [60]), and supervised classification. Pretrained models are obtained from the VISSL Model Zoo [68].

Data For our ImageNet experiments, we use the standard ILSVRC 2012 [163] training/validation split with per-channel normalization. We additionally in-

clude several datasets to evaluate the efficacy of our method in a self-supervised transfer learning setting. These datasets are CIFAR10/100, Street View House Numbers (SVHN), Food101, and EuroSat with splits as provided by VISSL, the same per-channel normalizations are used when inputting these images into the pretrained θ_j [103, 139, 18, 78]. No data augmentations are used in this work.

Training Adam optimizers with a learning rate of $3 \cdot 10^{-4}$ are used throughout all experiments with batch sizes of 4096 for all ImageNet training and 256 on all other settings [98]. For ImageNet experiments, 20 epochs are performed for both training and inference, otherwise, 50 epochs are used. In practice, we find it useful to warmup the MLPs (Φ) for half of the training epochs before allowing the learned representations to shift from their average initialization.

Evaluation We employ k-nearest neighbor (k-NN) evaluation throughout to measure the quality of features, qualitative comparisons remain constant under variance in the choice of k. The choice of k-nearest neighbor is made as it is a parameter-free evaluation method that requires minimal tuning. Under regularization-free linear evaluation transfer learning, our ensembled features outperform both our ensembled feature baselines as well as their component state-of-the-art transfer learning models (e.g. SimCLR). Under heavy regularization cross-validation as standardized in self-supervised learning works such as [72, 101], however, these gains are inconsistent (see Section 3.4.4).

Baselines We compare against several ensembling baselines throughout this work.

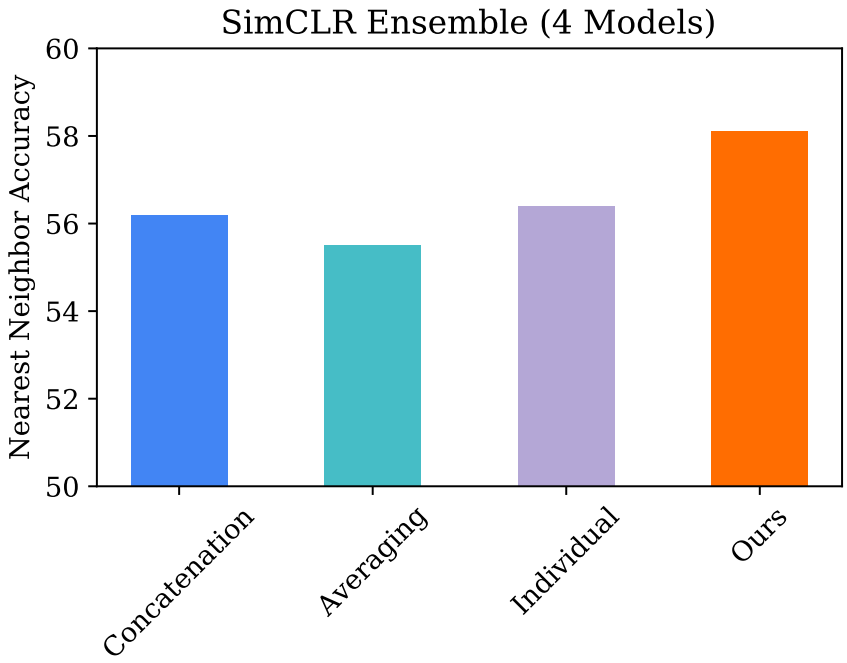

Figure 3.2: Nearest neighbor accuracies on the validation split of ImageNet. Our method improves over all baselines by over 2%.

- *Concatenation*: for image x_i the concatenation of all $z_i^{(j)}$ into a single vector $z_i^c \in \mathcal{R}^{2048m}$
- *Averaging*: the average feature $\frac{\sum_{j=1}^{m} z_k^j}{\|\sum_{j=1}^{m} z_k^j\|}$ (the initialization of our learned Ψ)
- *Individual*: a single model of the ensemble being used (in each case we detail specifically *which* model this is).

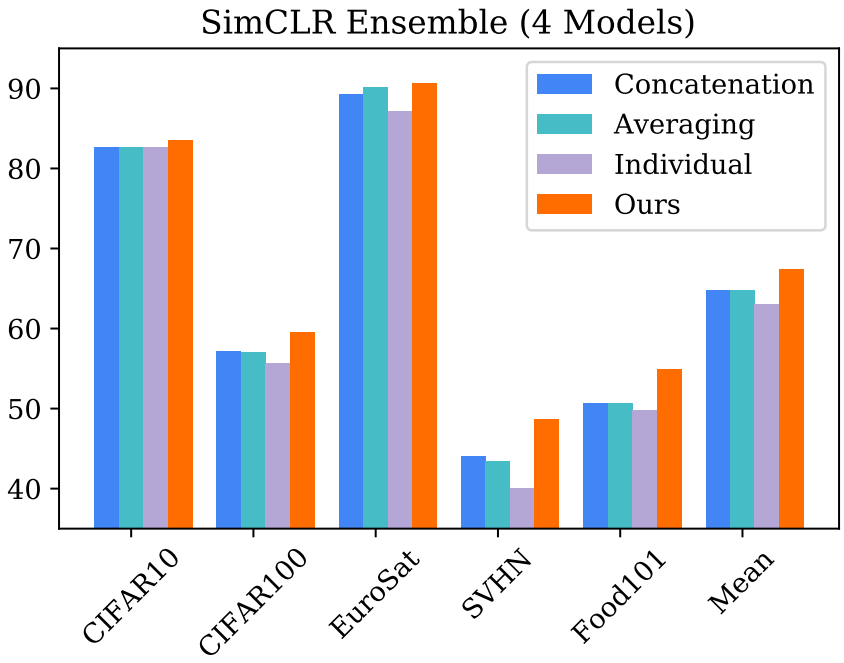

Figure 3.3: Our method applied to non-ImageNet datasets, leveraging the generalization of our pretrained feature extractors. Performance is improved across all datasets.

3.4 Results

3.4.1 Ensembling

In Figures 3.2 and 3.3, we use our method on an ensemble consisting of 4 SimCLR models. Figure 3.2 demonstrates the efficacy of our method on the *source* dataset, ImageNet. All of the ensembled models were trained in a self-supervised fashion on this source, but our method extracts an additional 2% of performance, increasing the nearest-neighbor accuracy to over 58%.

In Figure 3.3, learning is performed on novel datasets, this can be thought of as self-supervised transfer learning. Labels are not made available until eval-

uation time, and then only to measure the k-NN accuracy. Using the frozen SimCLR features extractors, our model learns representations which achieve over 2.5% higher k-NN accuracy on average, with a smallest win of 0.6% (over Averaging on EuroSat).

We employ different types of ensemble in Figures 3.4 and 3.5. Figure 3.4 employs an ensemble consisting of five differently trained self-supervised models: Barlow Twins, PIRL, RotNet, SwAV, and SimCLR. These represent various approaches to self-supervised learning: SwAV and SimCLR are more standard contrastive methods, while Barlow Twins achieves state-of-the-art performance using an information redundancy reduction principal. SwAV is a clustering method in the vein of DeepCluster [24] and RotNet is a heuristic pretext from the family of Jigsaw or Colorization [142, 243]. Here we use Barlow Twins as the "Individual" comparison as it notably achieves the highest individual k-NN accuracy on every dataset. This setting is intriguing from two different perspectives: firstly, the varying strengths of the underlying ensembled models is challenging as noisy signal from the weaker models can drown out that of the strongest; second, the varied pretraining methods results in different strengths. For example, RotNet is by far the weakest model of the ensemble with an average transfer k-NN accuracy almost 10% lower than any other model. However, on SVHN (a digit recognition task), it performs very well, beating all non-Barlow methods by over 4% (the efficacy of such geometric heuristic tasks on symbolic datasets as previously been noted in [202]). Our model seems to benefit from this, achieving its largest win over Barlow Twins (8.2%) on this dataset. This demonstrates the ability of our model to effectively include multiple varying sources of information.

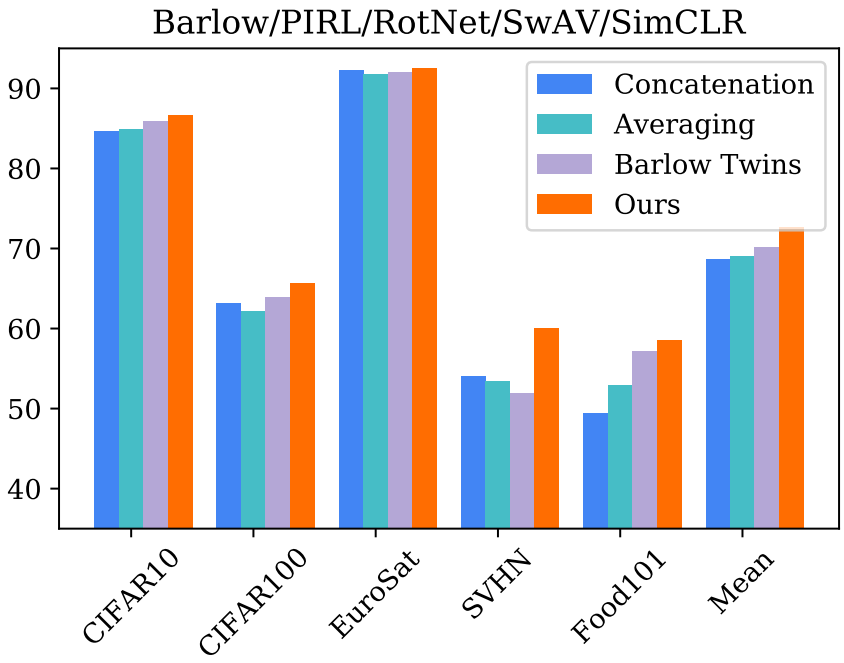

Figure 3.4: Ensembling varied models. We observe that our method seems to capture specialties/strengths of the component feature extractors, particularly the symbolic-dataset efficacy of RotNet.

In Figure 3.5, we consider the effect of using our method on a *supervised* ensemble. While the pretraining goals of these models are aligned and thus traditional techniques (e.g. prediction averaging) could be used, we demonstrate that our model successfully improves upon the ensembled intermediate features further demonstrating its agnosticity towards pretraining tasks.

3.4.2 Efficacy on Individual Models

While our method is designed as an ensembling technique, we discover that it is surprisingly effective when employed on *a single model*. These result are quite remarkable, as the improvement of features without access to their corre-

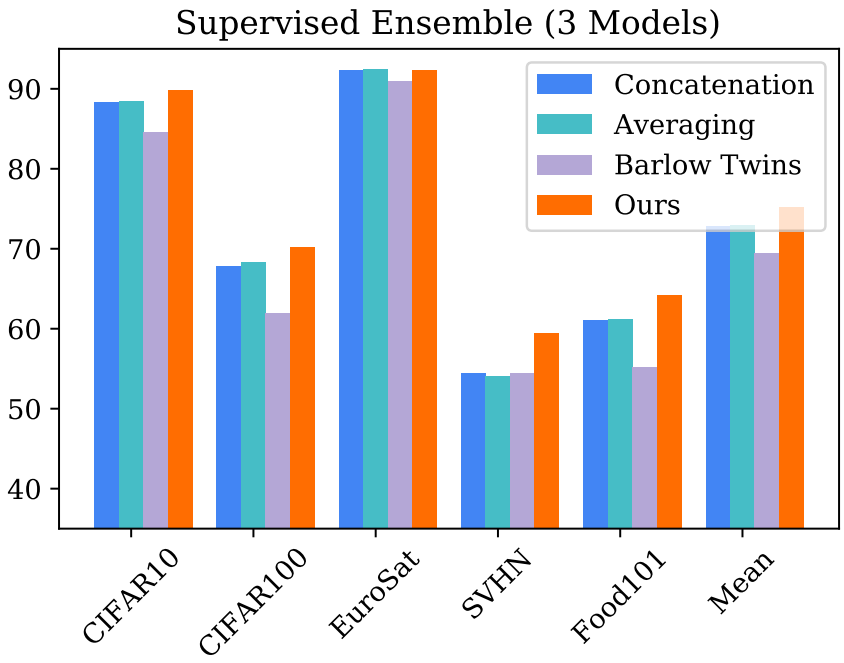

Figure 3.5: Despite not utilizing the consistency of the supervised classification objective, our method effectively combines supervised models to improve upon the performance of all other ensembles considered.

sponding images or additional supervision is quite challenging. This setting is especially remarkable as here *the input initialization and targets are identical*; we find that the MLP, ϕ, does not converge to a perfect identity function during the warmup period and the movement of the representations ψ in fact help enable near-perfect target recovery. In the inference stage, the MLP output of the average feature is close to identity (0.97 cosine similarity), but only by learning the representation through gradient descent does the similarity improve to near-perfect (0.99+) similarity. We discuss possible reasons for this behavior in Section 3.5.

In Figure 3.6, we see that our "ensembling" technique benefits all individual models substantially (1.8, 1.3 and 0.4% respectively) when our represen-

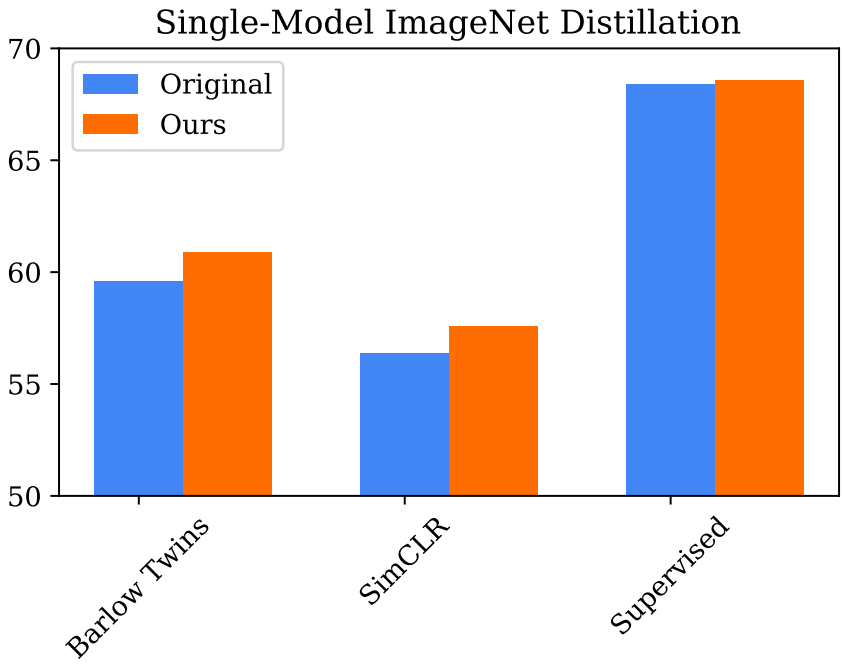

Figure 3.6: Paralleling the efficacy of self-distillation [241], our "ensembling" method proves to provide performance gains even when just one model is employed.

tations are trained on ImageNet. Given the degree to which the original self-supervised models' objectives are optimized, the margin of the improvement is quite impressive. This benefit carries over the self-supervised transfer learning as well (Figure 3.7). Here we use our method in conjunction with a Barlow Twins model. Our method offers a mean k-NN accuracy gain of over 1%, once again *despite no additional information, augmentations, or images being made available besides the CNN's features themselves*.

While this gain is relatively minor, we show that superiority to the baseline features is maintained across a wide range of hyperparameter choices. In Figures 3.8 and 3.9 we examine the robustness of our method (Single Barlow Twins Model on varied dataset benchmark) to choices of learning rate and batch size.

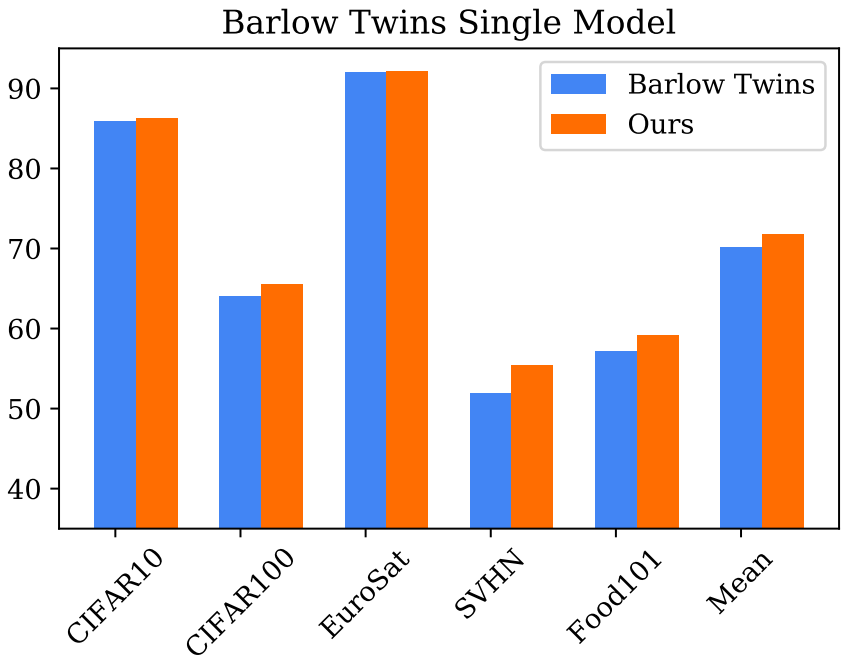

Figure 3.7: We experiment with a single Barlow Twins model on our generalization benchmark. Performance is gained on all datasets, with a mean improvement of over a full percent.

We find that superior performance to the baseline is maintained across a wide choice of settings (despite this being one of the settings where our margin of improvement is the smallest) and that there is in fact room for further per-task optimization via cross-validation of hyperparameters.

3.4.3 Transferring MLPs from ImageNet

So far we have only considered the scenario where the MLPs, Φ, are trained on the same dataset as the representations Ψ, where inference is ultimately performed. This is not a necessary assumption of our framework, however, once MLPs are trained on a dataset they can be re-used to learn representations Ψ

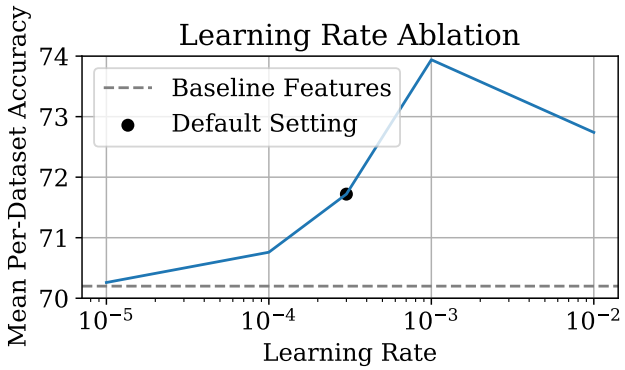

Figure 3.8: Mean per-dataset accuracy, single Barlow Twins model on varied dataset benchmark. Learning rate ablation.

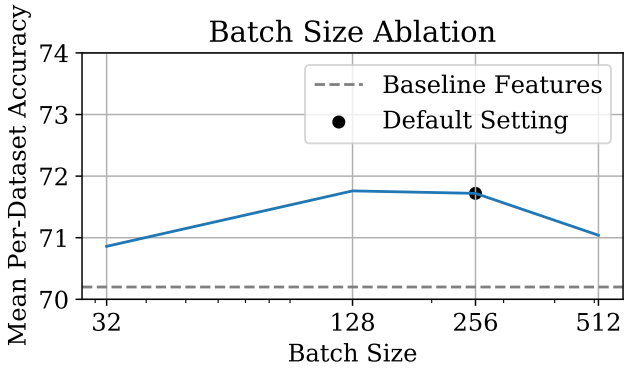

Figure 3.9: Mean per-dataset accuracy, single Barlow Twins model on varied dataset benchmark. Batch size ablation. Linear learning rate scaling rule followed.

on arbitrary imagery. We conduct experiments where Φ is trained on ImageNet (specifically those generated for use in Figures 3.2 & 3.6) are re-used in the *transfer* setting. Because the MLPs are frozen, no parameters of any networks are being changed during training, solely the representations Ψ are being learned. The results of these experiments are shown in Figures 3.10 & 3.11.

In the ensemble setting, the performance is largely maintained when re-using MLPs from ImageNet. Both method (*Ours (Standard)* and *Our (Transfer*

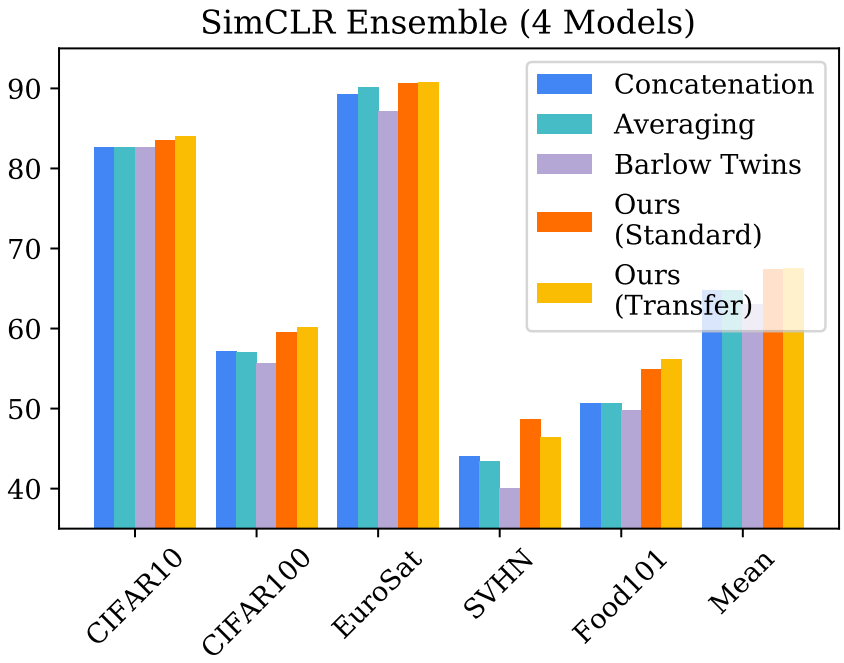

Figure 3.10: The MLPs do not necessarily need to be trained on the target dataset, but can be transferred from ImageNet to other downstream datasets. Here no parameter tuning is done on each dataset, gradients are computed solely to directly learn the representations. We find in fact that this transfer approach maintains the performance of directly learning new MLPs.

MLPs)) are still superior to the baselines across *all* datasets. This result is intriguing, as the observed ensemble gains are relatively large, and this method requires *no* pretraining on a targeted transfer set, simply inference via gradient descent of Ψ; such generalization properties are a part of what makes the base pretrained models so valuable in application.

In the single-model setting, the Barlow Twins model + MLP trained on ImageNet is re-used across transfer datasets. Here we see a notable decline in performance, but the transferred model still maintains improvement over the baseline on 4 out of 5 datasets (all but EuroSat).

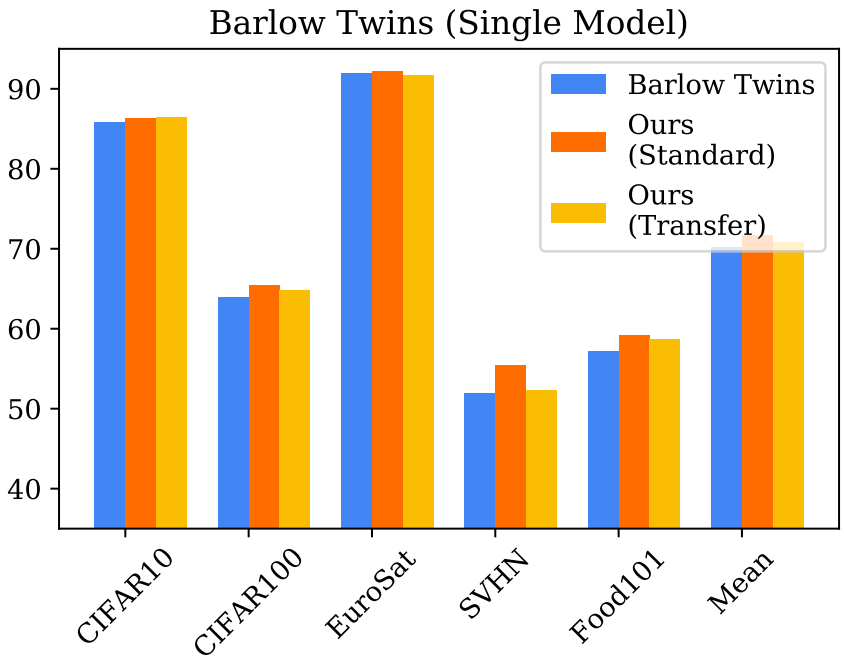

Figure 3.11: In the single-model case, transferring ϕ still provides benefit over the baseline, but is less effective than learning the MLPs per-dataset.

3.4.4 Cross-Validated Linear Regression

We follow a protocol similar to [72, 101]. A validation subset (10% of the training set) is sampled and held out during training. Training on the remaining 90% of the data is performed for a hyperparameter sweep with performance on the validation subset being measured. We employ an SGD optimizer for 1000 epochs with a batch size 4096 and learning rate 1.6. A weight decay sweep of $\lambda \in \{1e-6, 1e-5, 1e-4, 1e-3, 1e-2\}$ is performed. Results are shown in Figure 3.12. We see that our method in the single-model setting suffers substantially compared to the baseline. In the ensembled setting, there is still improvement on-average, but the gains are much more inconsistent than under k-NN evaluation. While this is a current limitation of our model the benefits under

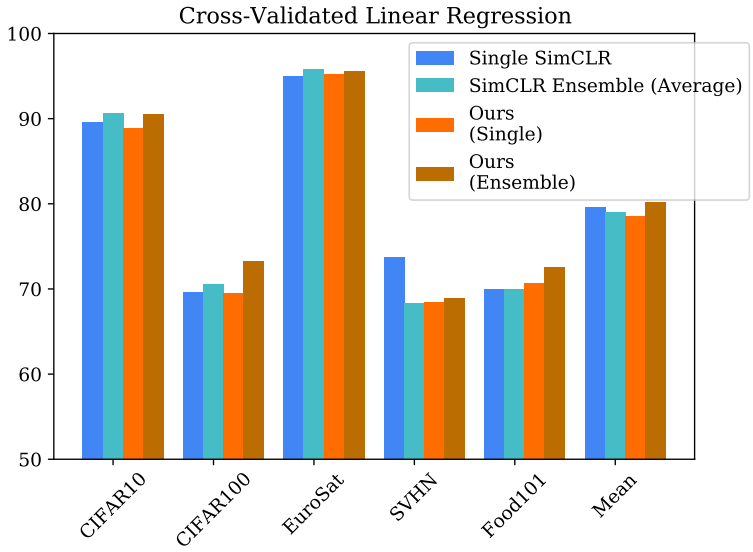

Figure 3.12: Linear regression accuracies with cross-validated L2-regularization. Relative performance of our method is worse compared to the k-NN evaluation setting.

k-NN indicate a fundamental utility in our ensembling method to learn new reprsentations.

3.5 Analysis

3.5.1 Deeper Φ Regularize Ψ

One hypothesis for the efficacy of our method in the single-model setting is that ϕ acts as a regularizer. The findings of [8] demonstrate that stochastic gradient descent in deep neural networks tends to recover low-rank solutions when performing matrix factorization, this helps to explain the generalization properties of deeper networks: deeper networks lead to simpler solutions which tend to generalize better. This relates to our method, as our improvements hinge upon

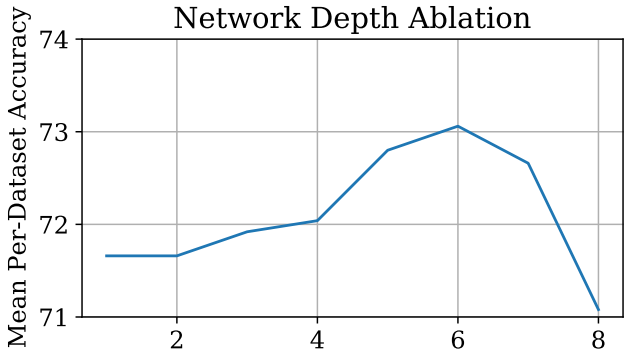

Figure 3.13: Ablation of MLP depth, possibly suggesting that the low-rank tendency of deeper networks serves as a regularizer on the learned representations. This results in improved representation quality with network depth up to 6 layers.

the network ϕ *not* learning a perfect identity function during warmup; if it did, then the gradients with respect to Ψ would vanish and no change of representation would occur. [8] suggests then that a deeper network might improve the ultimate quality of Ψ, as the points would need to be recoverable by a lower-rank MLP.

We confirm this phenomenon in Figure 3.13 by varying the depth of Φ from 1 to 8 layers while learning representations directly on our varied dataset benchmark using a Barlow Twins model. Increasing depth improves accuracy incrementally over a total of '% on average until the network is 6 layers deep, more than triple that of our default setting. Some but not all of this performance boost is recoverable by adding in small amounts of traditional weight decay (e.g. $1e-6$) to the parameters of the MLP.

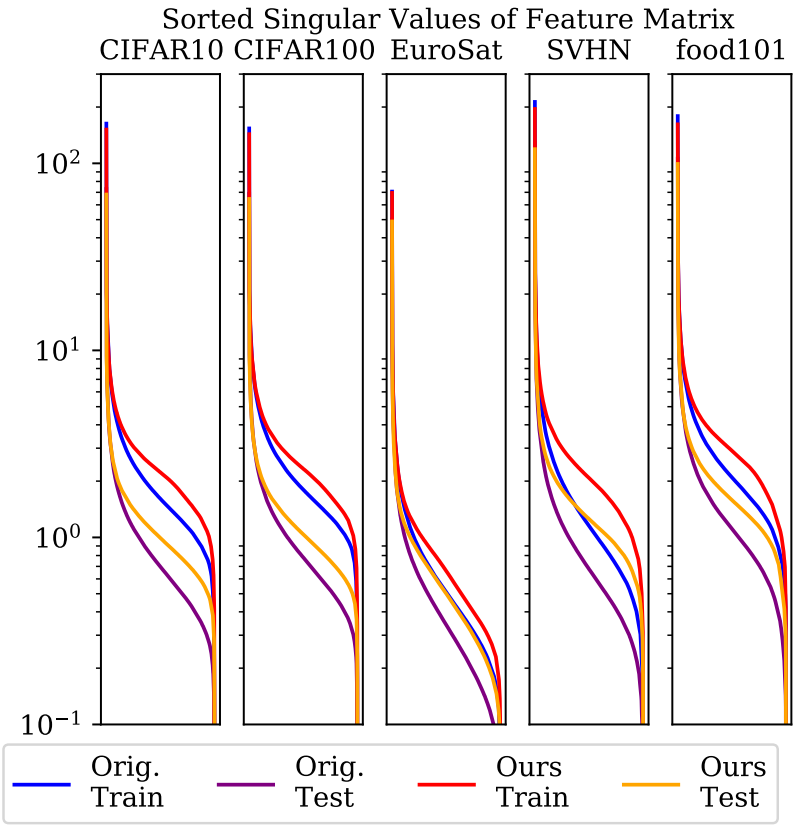

Figure 3.14: Sorted singular value curves for our method vs. the baseline features in an apples-to-apples setting (learning ϕ restricted to non-negative). Our method learns features with a more balanced set of singular values, indicating a more uniformly spread bounding space.

3.5.2 Behavior of Model

Now that we have established a partial explanation of why our model works, by learning representations which preserve information under the regularization of an SGD-learned deep network, we now investigate what specific changes our method makes to the feature space.

First, in Figure 3.14, we examine the distribution of Ψ. We do so by training

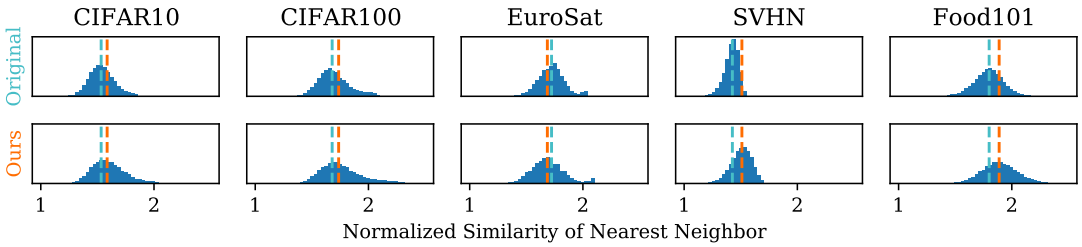

Figure 3.15: The normalized maximum similarity (measure of how close each test point's nearest neighbor in the trainset is relative to average) for each method-dataset pair. Dashed lines indicate the median for each distribution We see that the proposed approach generally has higher normalized maximum similarities, indicating relatively tighter clustering behavior.

representations from a single Barlow Twins model similar to previously, with the important distinction that we restrict our points to be non-negative (i.e. in the first n-tant of feature space), to make an apples-to-apples comparison to the baseline features. We examine the singular values of this (constrained) feature matrix compared to that of the original features. In general, the singular value distribution of Ψ are less heavy-tailed: meaning the volume occupied by the features is larger and more uniform in each dimension than the baseline features. This is an indication of Ψ learning a regularized form of the original Z. The above finding indicates that the feature representations are spread out as a result of the learning process.

We also investigate what happens to clusters in Figure 3.15. Here we wish to investigate what happens to clusters. We examine this again through the lens of nearest-neighbors: for each ψ'_k in the test set, we calculate the maximum cosine similarity of points in the train set. This maximum similarity is then normalized by the *mean* similarity for each method-dataset pair. The resulting normalized similarity provides a measurement of how relatively close points are to their

nearest neighbor vs. an average pair of points (with a higher value indicating relative closeness). Histograms of this metric are shown in Figure 3.15. For 4 out of the 5 datasets, our method results in tighter neighbor matchings than the baseline. Intriguingly, the one dataset for which this does not hold is EuroSat, is also the dataset where our models consistently yielded the lowest benefit. These findings suggest that our methods success is partially attributable to accentuation of existing clusters in the dataset. It is interesting to note as an aside that the mean pairwise similarity across the entire dataset is significantly lower for our method, cementing the findings from Figure 3.14 that the features are more distributed across space. In summary, the findings suggest that our methods success is partially attributable to accentuation of existing clusters in the dataset.

3.6 Discussion

In this work, we presented a novel self-supervised ensembling framework which learns representations directly through gradient descent. The intuition behind our method is to capture all of the knowledge contained in the ensembled features by learning a set of representations from which the former are fully recoverable. We demonstrated the efficacy of our method in Section 3.4 and analyzed causes and effects of the representation improvement in Section 3.5. We hope that this work lays the groundwork for further forays into the problem of utilizing combinations of the powerful pretrained models that are becoming plentiful in the computer vision literature.

As previously noted, while we demonstrate improvement under k-NN in

this paper the average feature baseline surpasses our method under linear evaluation when regularization is heavily optimized under a grid sweep as standardized in [72, 101]. We tried applying various regularizations during MLP training, including traditional L2 weight decay, L1 regularization of Ψ, the dimensionality of Ψ, and the depth/width of the MLPs Φ. While some of these modifications further increased the k-NN performance improvements, when representations were evaluated under linear regression with cross-validated regularization none consistently surpassed the average ensemble baseline.

CHAPTER 4

CAN WE CHARACTERIZE TASKS WITHOUT LABELS OR FEATURES?

4.1 Introduction

Transfer learning is key to the success and popularity of computer vision. The features from a convolutional neural network (CNN) trained on one task can be incredibly useful across a broad variety of tasks[202, 181, 192, 114]. Even larger performance gains can be obtained by *selecting* a pretrained model more specially suited to the task at hand. This behavior has been studied in past work[39], but only recently has attention turned to *how to determine* which specialized model is appropriate for a given task[4].

The key to such model selection is characterizing tasks and their relationships. What does one need to characterize a task? *A priori*, it seems that we need to characterize two things:

1. The *input*, characterizing which requires **a dataset** of images, and **features** to represent them, and
2. The *output*, characterizing which requires **labels** for the dataset.

Current approaches to characterizing tasks synthesize both sources of information. Task2Vec[4], for example, trains a linear head on top of a pretrained feature extractor and uses the Fisher information associated with the resulting model to generate a vectorized embedding. Decisions such as choosing the best pretraining task for a target task can then effectively be made using retrieval-like techniques with this embedding.

But how much of these accurate decisions come from characterizing the *input domain alone*, and how much comes from knowledge of the precise task? This question has important practical considerations. For example, suppose we want to choose a pretrained representation for analyzing x-ray images. We may not yet know *what* we want to recognize in x-ray images. In fact, we may want a pretrained representation suitable for *any* kind of x-ray image analysis, even those we haven't conceived yet. In such cases we are interested in characterizing only the general problem domain, and do not have particular labels (yet) that we are interested in. This raises the question: **Can we characterize tasks without labels?**

As our first contribution, we answer this question in the affirmative. To address this problem, we introduce PseudoTask: a modification of the Task2Vec algorithm that replaces labels with *pseudolabels* output by an image classifier trained in a generic source domain. While these pseudolabels are definitely incorrect due to domain misalignment, they prove discriminative enough to be quite useful in characterizing tasks. Empirically, we find that PseudoTask embeddings are *as accurate as supervised Task2Vec embeddings*, indicating that one can characterize tasks effectively *even without labels*.

Key to this performance, as also to the performance of Task2Vec, is the inductive bias provided by the pre-trained feature representation. However, this inductive bias may prove harmful as the task domains move farther away from the domain where the feature extractor is pretrained [202], making it risky to rely so heavily on such feature representations. This raises a second question: **can we characterize tasks without features**?

We answer this too in the affirmative. We design a method called Task Tan-

gent Kernel (TTK) that measures task similarity using the gradients of randomly initialized networls. TTK *does not use pretrained probe networks at all*. Despite this lack of inductive bias, it provides useful selections as well, at less than half the error of PseudoTask and Task2Vec with random feature extractors.

In sum, this paper introduces two new techniques for characterizing tasks that lift some of the restrictive assumptions of prior work (Figure 4.2):

1. We introduce **PseudoTask**, a new way of characterizing tasks without labels, allowing one to characterize problem domains in general. We find PseudoTask performs almost the same as Task2Vec, indicating that labels are in fact not necessary.
2. To avoid the potentially mismatched inductive bias of pretrained feature extractors, we introduce **Task Tangent Kernels**, which characterizes tasks effectively even without such feature extractors.

4.2 Related Work

Previous work has studied why pretrained models transfer so well[245, 101], the tradeoffs between specialization and scale in pretraining[39] and how concurrent multitask learning can benefit performance[235]. Task2Vec[4], the work that this paper builds off, studies the problem of automated model selection, as does [86]. [172, 2, 109] recommend *algorithms* for various problems using "Active Testing", intelligently adapting exploration based on results. Such types of approaches are inherently more limited computationally than embedding-based methods. [212] predicts per-image performance for single models, while

[55, 140] characterize performance per-dataset. [126, 242] deal with the problem of model recommendation for action recognition and object detection respectively, but require some degree of performance evaluation on the new task in order to provide a recommendation.

Transfer learning has been increasingly optimized, with recent advances detailed in [99]. The problem of expert selection has analogs to image retrieval, examples of which include[162, 10]. Other works measuring distances between domains include evaluating the biases between semantically similar datasets[186] and measuring temporal domain shifts in datastreams[97]

Our PseudoTask framework draws heavily on pseudolabeling for semi-supervised learning, such as in [108]. Parts of our training setup are very similar to that of self-training, such as for few-shot transfer or semi-supervised learning[153, 223]. PseudoTask can be considered a form of self-supervised training such as [60, 142, 29, 72, 76]. The Task Tangent Kernel is inspired by Neural Tangent Kernel literature, originating with [83] and developed in [112, 9, 143].

4.3 Problem setup

A **task** $T = (X_T, Y_T)$ consists of a **domain** of images $X_T = \{x_i\}_{i=1}^{n_T}$ and corresponding **labels** $Y = \{y_i\}_{i=1}^{n_T}$. An **expert** $\Theta_T = (\phi_T, h_T)$ is a dedicated neural network, consisting of a feature extractor ϕ_T and a head h_T that is trained on a task T. Suppose that we have a bank of tasks $T_1, \ldots, T_N$, and have already trained a corresponding bank of experts $\Theta_{T_1}, \ldots, \Theta_{T_N}$. Then, when we encounter a new task T', instead of training a model from scratch, we may want to choose a pretrained

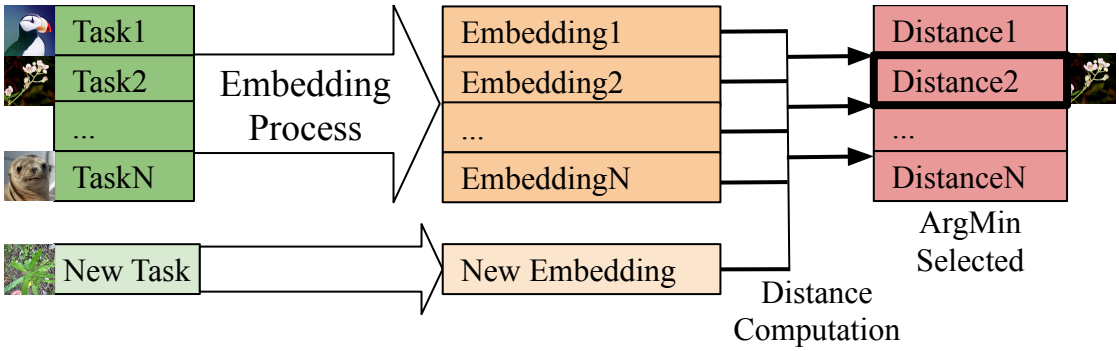

Figure 4.1: Tasks consisting of images and possibly labels are embedded into a vector space. When a new task is introduced, a new embedding is calculated and compared (using a modified cosine distance) to the bank of previous embeddings. The closest embedding is selected to use as pretraining for the new task.

expert Θ_{T_i}, use its fixed feature extractor ϕ_{T_i} and train a linear head to solve the new task; see Figure 4.1. *The goal of expert selection is to pick the best pretrained expert for the target task.*

4.4 Background: Task2Vec

Our work builds on Task2Vec[4], a recent approach that tries to characterize tasks and embed them in a useful way. Task2Vec tries to characterize both the input domain, as well as the semantic information carried by the labels. The key intuition behind Task2Vec is that we can try to solve the task with a moderately effective but generic off-the-shelf feature extractor (e.g., trained on Imagenet) and then see which parameters of the feature extractor most impact the performance. Task2Vec posits that tasks which are sensitive to the same set of feature extractor parameters are likely to be "similar" to each other, especially in terms of what they demand out of pretrained features.

Concretely, Task2Vec trains a linear layer on top of the off-the-shelf feature extractor (called a "probe") for the task in question. It then computes the *Fisher Information Matrix* (FIM), which is known to measure the sensitivity of the loss to the parameters of the model. Denoting by $p_w(y|x)$ the output distribution of trained model p_w with weights w, and by $\hat{p}(x)$ the data distribution, the FIM is defined as:

$$F = \mathbb{E}_{x,y\sim\hat{p}(x)p_w(y|x}[\nabla_w \log p_w(y|x)\nabla_w \log p_w(y|x)^T] \tag{4.1}$$

Task2Vec estimates F using a variational approach that amounts finding the optimal weights $\hat{w}$ and precision matrix Λ that minimize the following objective:

$$\begin{aligned} L(\hat{w}, \Lambda) =& \mathbb{E}_{w\sim\mathcal{N}(\hat{w},\Lambda)}[L_{CE}(X_T, Y_T, p_w)] + \\ & \beta KL(\mathcal{N}(0,\Lambda)||\mathcal{N}(0,\lambda^2 I) \end{aligned} \tag{4.2}$$

Here L_{CE} is the cross entropy loss over the dataset and λ is a hyperparameter. Achille et al. prove that the solution to this is $\Lambda = F + \frac{\beta\lambda^2}{2N}I$. Task2Vec finally takes the diagonal of F and averages together the values for different parameters of the same filter to produce the embedding.

When using this embedding, symmetric distances such as cosine distance, denoted d_{sym}, do not yield satisfactory pefrformance when retrieving experts. This is because there is an inherent asymmetry to the expert selection problem: a task with a large dataset and thousands of classes will yield a good expert for a similar task with only two classes and a small dataset, but not vice versa. Therefore, there is a large benefit to making the distance function *asymmetric* to account for the complexity of the task. The Asymmetric Task2Vec distance (d_{asym}) is defined as:

$$d_{asym}(t_A \implies t_B) = d_{sym}(t_A, t_B) - \alpha d_{sym}(t_A, t_0) \tag{4.3}$$

Here t_0 is the "trivial" task of ImageNet classification. This formulation makes complex tasks that are very different from ImageNet relatively closer to everything else. The intuition is that a more complex task has a higher chance of being a relevant expert given the same degree of symmetric similarity. α's value varies by architecture. In the original work, $\alpha = 0.3$ is reported as optimal when training with a ResNet-34[77]. In our experiments with ResNet-18s we found $\alpha = 0.15$ to yield best performance. See Section 4.8 for further discussion of α.

Limitations: The Task2Vec formulation requires a fully specified task to have a labeled dataset, as well as a pretrained probe network to be available. We next address these limitations using our proposed alternatives below.

4.5 Characterization Without Labels: PseudoTask

Motivation: The first step in Task2Vec is to train a linear classifier with a probe network's features for the task in question. This uses labels, which in turn provide the semantics of the task. But often we may want to characterize entire problem domains (e.g., x-ray images) without having a specific task in mind. In such cases, labeled datasets may be unavailable.

Even for well-specified tasks, the amount of labeled data required to characterize the task using the Task2Vec approach can also be substantial, precluding applications like few-shot learning where one may want to choose experts and take decisions with only one or two labels per class. To alleviate this issue, we present a self-supervised task characterization, which we dub "PseudoTask".

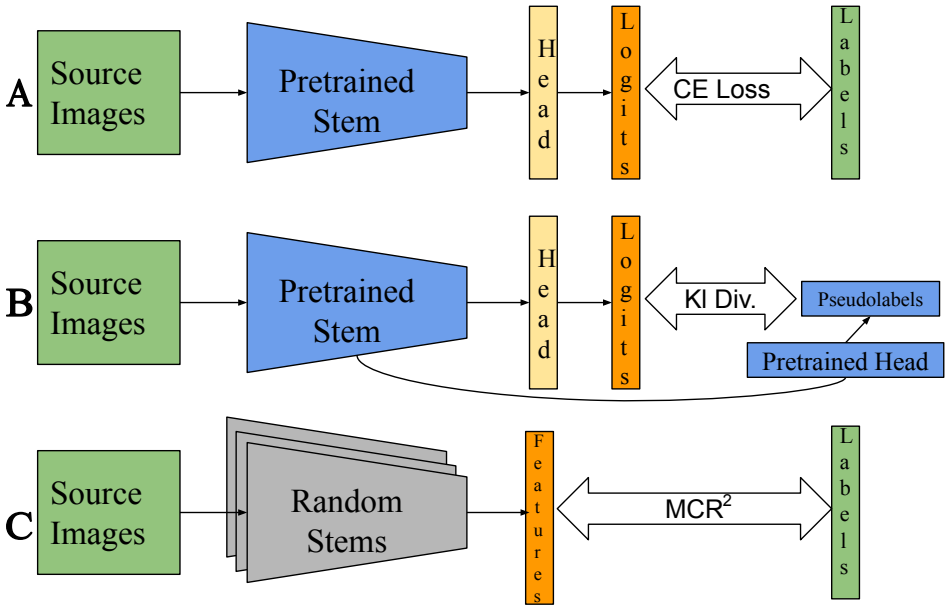

Figure 4.2: **A**: The original Task2Vec[4] framework. A pretrained model is available as are image labels, which a linear head is trained to predict. **B**: Our proposed PseudoTask framework. Pretrained models are available, but labels are not. A zero-initialized head is trained to match the predictions of the full pretrained network. **C**: Our proposed Task Tangent Kernel framework. Labels are available, but pretrained models are not. No training is done. Gradients are calculated from the features and labels using Maximal Coding Rate Reduction (MCR^2)[233] across randomly initialized networks (see Sec. 4.6).

4.5.1 Method

What information can we use to characterize a problem domain with no labels? Self-training approaches have recently shown the usefulness of training a network in one domain to match the predictions of a teacher from a source domain *even when the two domains share no classes at all*[153, 223]. Furthermore, unsupervised contrastive learning approaches such as MoCo and SimCLR[76, 29], demonstrate the emergence of semantics simply by learning to distinguish individual images.

Both approach types point to the striking power of *pseudolabels*, and motivate us to create PseudoTask, where pseudolabels are used as a substitute for ground-truth labeling. Concretely, PseudoTask follows the originally presented framework of Task2Vec with a key modification: instead of real labels, we use soft labels from a pretrained classifier (ImageNet or Places365).

1: Compute soft labels $\ell_i = \Theta(x_i)$ for each image
2: Zero out the linear classification layer (γ)
3: **for** 2 epochs **do**
4: Fit the linear head γ to $\{\ell_i\}$
5: **end for**
6: **for** 10 epochs **do**
7: Minimize Task2Vec loss: $L(\hat{w}, \Lambda) = \mathbb{E}_{w \sim \mathcal{N}(\hat{w}, \Lambda)}[L_{CE}(x', \Theta(x'), p_w)] + \beta KL(\mathcal{N}(0, \Lambda) \| \mathcal{N}(0, \lambda^2 I))$
8: **end for**

Algorithm 2: PseudoTask

Given a domain (task) $T = (X_T)$ consisting of unlabeled images $X_T = \{x_i\}_{i=1}^{n_T}$ and a pre-trained classifier Θ, we follow Algorithm 2 to compute our embedding. Note that when computing the Task2Vec embedding (second for loop), the soft predictions are computed dynamically on *randomly augmented* versions (x') of the images (x).

Asymmetric PseudoTask: The final adjustment we found necessary in the self-supervised algorithm was the measure of asymmetery. In the original work, a bias term of $-\alpha d_{sym}(t_{source}, t_0)$ was used where t_0 is the "trivial" ImageNet task. While the benefit of this term for Task2Vec was replicated in our baseline experiments, we found that it was not appropriate for PseudoTask. For PseudoTask, the distance bias term $d_{sym}(t_a, t_0)$ is fairly homogeneous across tasks compared to the variation for embeddings trained with Task2Vec. We hypothesize that this behavior stems from the training objective of PseudoTask being consistent

across domains (matching soft labels from the same pretrained classifier) while Task2Vec varies more significantly in label distribution.

To define an alternative asymmetric distance, we leverage the observation that a large norm of the PseudoTask (and Task2Vec) embedding is correlated with task hardness: for a complex task, linear classifiers on the probe feature extractor will not work well, yielding large-valued Fisher Information Matrix (see Sec. 2.2 in [4]). As such, to bias the expert selection towards experts trained on more complex tasks, we add a bias term based on the norm of the embedding and define PseudoTask's asymmetric distance between two task embeddings, t_a and t_b as:

$$d'_{asym}(t_a \implies t_b) = d_{sym}(t_a, t_b) - \alpha ||t_a|| \tag{4.4}$$

The intuition behind this definition is similar to the Task2Vec asymmetric distance, but uses a slightly different formulation.

4.6 Characterization Without Features: Task Tangent Kernel

Motivation: Task2Vec and PseudoTask both use pretrained feature representations. Even for Task2Vec, which does not need pseudolabels, the availability of this feature representation is critical. Task2Vec relies on the Fisher information matrix, which is typically used to characterize how sensitive the *optimum* parameter setting is. If the feature extractor is far from optimal, using the Fisher information does not make sense. Unfortunately in practice one may operate in such drastically different domains that a given pretrained feature extractor is no longer optimal. Indeed, we find that if the feature extractor is far from optimal, Task2Vec fails at effective characterization (Sec.4.7.4). We therefore need

an alternative approach that does not rely so heavily on a suitable feature representation.

The derivatives of random neural networks We want to declare two tasks to be similar if and only if a model trained on one produces a good feature extractor for the other, or alternatively, the optimal feature extractors for the two tasks are close to each other. A brute force approach to measuring task distance might thus be to separately train models for each task from the same initialization and look at how far the optima are in parameter space.

Of course, this is prohibitively expensive and obviously defeats the point, since we wanted to avoid training a separate model for the target task anyway. But what if we don't train these models the whole way? Can we instead just train these models for very few epochs or steps and then evaluate how far they are? Concretely, imagine we start the training for both tasks using the same initialization, and take a single step. If the optimal models for the two tasks are close to each other, one might imagine that the very first update will also be close. If we repeat this for multiple initializations and find that the first update for the two tasks are always close, then one might conclude that the tasks are "similar". Standard optimization procedures rely on gradient descent, so this first update corresponds to the gradient of the randomly initialized model. Thus, we hypothesize that a measure of task similarity could be computed by calculating the *expected similarity between the gradients of the tasks at the same random initialization*.

The Neural Tangent Kernel, which was first proposed in [83], describes the convergence behavior of neural networks in the limit of infinite width. A side-

effect of this analysis is a *kernel function* between data points that comes close to mimicking the behavior of trained neural networks, but *itself requires no training*. This kernel takes the form:

$$k(x, x') = E_\theta\langle\frac{\partial f(\theta, x)}{\partial \theta}, \frac{\partial f(\theta, x')}{\partial \theta}\rangle \tag{4.5}$$

where x and x' are data points (e.g., images), and θ is the parameters of a randomly initialized neural network drawn from a fixed distribution (typically Gaussian).

Our work essentially adapts the above kernel to operate on *tasks* instead of *points*. Based on the results with NTK, we reason that computing the NTK kernel over tasks instead of individual data points (by simply averaging the gradients of points in a task) thus provides a measure of similarity between the tasks.

We concretize this intuition as follows. Suppose we are given two tasks $T_1 = (X_1, Y_1)$ and $T_2 = (X_2, Y_2)$ consisting of images $X_k = \{x_i^{(k)}\}_{i=1}^{n_k}, k = 1, 2$ and corresponding labels $Y_k = \{y_i^{(k)}\}_{i=1}^{n_k}, k = 1, 2$.

Suppose L is a loss function such that given a feature extractor ϕ, $L(\phi, X, Y)$ measures how well the feature extractor is able to separate out the classes in the task (X, Y). We randomly initialize N feature extractors $\{\phi_i\}_{i=1}^N$ with parameters $\{\theta_i\}_{i=1}^N$. For each feature extractor, for each task, we compute the gradient of the loss with respect to the feature vector parameters:

$$g_i^{(k)} = \frac{\partial L(\phi_i, X_k, Y_k)}{\partial \theta_i} \tag{4.6}$$

We then define a *kernel* between the two tasks as the average cosine distance

between the two gradients :

$$k(T_1, T_2) = \frac{1}{N} \sum_{i=1}^{N} \frac{\langle g_i^{(1)}, g_i^{(2)} \rangle}{||g_i^{(1)}|| ||g_i^{(2)}||} \tag{4.7}$$

Because we are computing this task kernel using the gradients, we call this kernel the *task tangent kernel*. We only use the last residual block of the ResNet to compute the embedding as it contains the most channels.

Loss function:A key component here is the loss function L. Typical loss functions such as cross entropy operate on predictions rather than features, which presents difficulty given the permutation-variant nature of a randomly initialized classifier head. We use the Maximal Coding Rate Reduction loss(MCR^2)[233]. This loss measures how close same-class features are, and how spread out the dataset is in feature space. Specifically, we embed the images X using the feature extractor ϕ yielding embeddings $Z \in \mathbb{R}^{d \times m}$ (d being the feature dimensionality, m the total number of data points). Let the set of embeddings of class j be denoted by Z_j. Then the loss is defined as:

$$L(\phi, X, Y) = -R(Z) + R_{class}(Z) \tag{4.8}$$

$$R(Z) = \frac{1}{2} \log \det \left(I + \frac{d}{m\epsilon^2} ZZ^\top \right) \tag{4.9}$$

$$R_{class}(Z) = \sum_j \frac{m_j}{2m} \log \det \left(I + \frac{d}{m_j \epsilon^2} Z_j Z_j^\top \right). \tag{4.10}$$

With m_j as the number of data points with membership in class j and ϵ a "prescribed precision" constant ($\epsilon^2 = 0.5$ in our work, see [233] for details). The functions R and R_{class} describe the whole-dataset and per-class *coding rates*, measuring the compactness of all or subsets of features. This loss encourages the dataset as a whole to be non-compact (discriminable) while the class subsets should be highly compact (clustered), bearing resemblance to the supervised

contrastive learning objective such as in[96].

4.7 Experiments

4.7.1 Meta-Task and Baselines

We perform experiments on the **CUB+iNat** meta-task of the Task2Vec paper [4], denoted as $\mathcal{T}$. This set consists of 25 species classification tasks from Caltech-UCSD Birds[215] and 25 from iNaturalist[195]. Tasks are sets of species grouped at either the Order or Family level. For each individual task $T \in \mathcal{T}$, the benchmark requires us to choose an expert from $\mathcal{T} - \{T\}$. We measure the error obtained with this choice, relative to the error of the optimal choice, reporting the average relative error across all tasks.

Baselines: The **Random** baseline selects a task from $\mathcal{T} - \{T'\}$ uniformly at random, and uses the corresponding expert. **ImageNet Initialization** does not use any of the experts available in $\mathcal{T}$, but instead uses a pretrained ImageNet network every time (an option not available to selection algorithms). **Average Features** uses an ImageNet-pretrained feature extractor to compute the average feature vector of images in the task, which is used as a task embedding (under cosine distance). Other metrics based solely on ImageNet-pretrained features, such as the H-Divergence[97] (accuracy of a linear classifier separating domains) between tasks produced similar or worse results. **RSA** and **LEEP** are more sophisticated techniques that analyze the features produced by each expert-target pair. While these techniques prove to be quite effective, the computational cost of running inference using each expert quickly can become pro-

Method	Error Increase	Labels?	Pretraining?
LEEP [140]	20.8%	✓	✓
RSA [55]	8.8%	✓	✓
Random Selection	59.5%		
EMD [40]	51.3%	✓	✓
Average Features	39.2%		✓
ImageNet Init.*	30.2%		✓
Task2Vec (Rand. Init)	48.7%	✓	
Task2Vec (Orig)	8.9%	✓	✓
Task Tangent Kernel	21.4%	✓	
PseudoTask (ImageNet)	20.4%		✓
PseudoTask (Places365)	10.0%		✓

Table 4.1: Metric reported is the mean increase of relative error between a method's choice and the optimal, averaged across the 50 tasks of **Cub+iNat** from [4]. Gray indicates methods which require running inference with each proposed expert on the target dataset which quickly becomes computationally prohibitive. Both TTK and PseudoTask outperform all baselines that do not require running inference from each expert on the new dataset. Furthermore, PseudoTask using Places365 initialization almost equals supervised performance.

hibitive as the number of considered tasks increases.

4.7.2 Main Results

We present our main findings in Table 4.1 and Figure 4.3. Both proposed methods outperform baselines, with PseudoTask achieving performance on par with Task2Vec. Note that LEEP and RSA require computing predictions for each expert/task pair; in the case of RSA an entire model must be *trained on the target task* before the expert initialization is chosen. In contrast, embedding methods

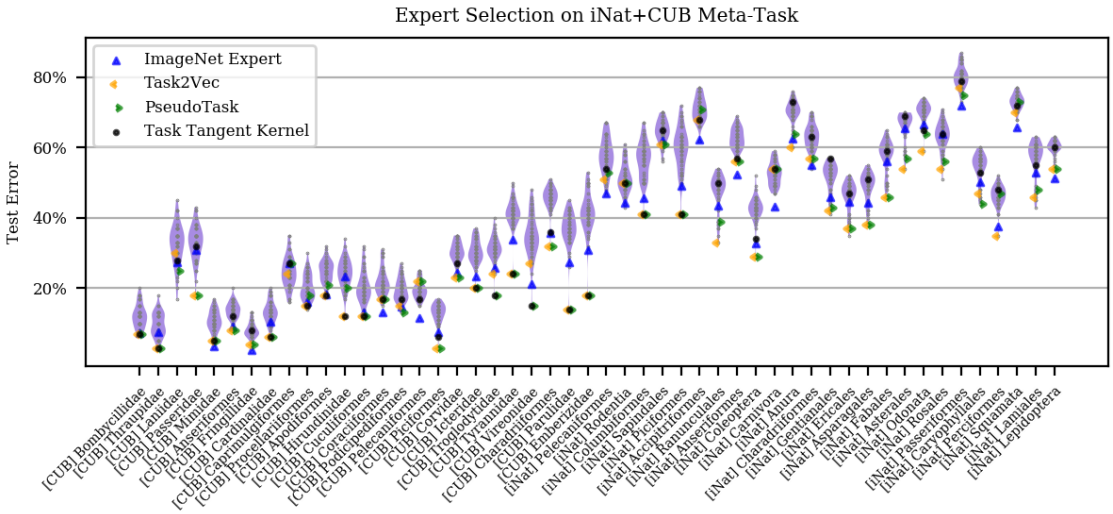

Figure 4.3: Violin plot of the error on each target task (x-axis) obtained by training a linear head on top of a model from each of the other 49 tasks. Markers indicate selection algorithm choices. Both of our methods (PseudoTask and Task Tangent Kernel) reliably outperform using an ImageNet feature extractor. The experimental setting is the same as Figure 3 of [4].

only require per-pair vector arithmetic and representations are persistent.

Task Tangent Kernel TTK has the most significant handicap, operating solely on the provided task dataset as opposed to other methods which employ networks pretrained on over a million images. Despite this, TTK is still useful, beating the strong baseline of initializing from ImageNet every time (ImageNet is not a permitted expert to select in the benchmark). We see in Section 4.7.4 that the performance by TTK is *far* superior to any other method operating off of random initialization and that this benefit stems from using *multiple randomized* networks, in accordance with theoretical work involving the Neural Tangent Kernel.

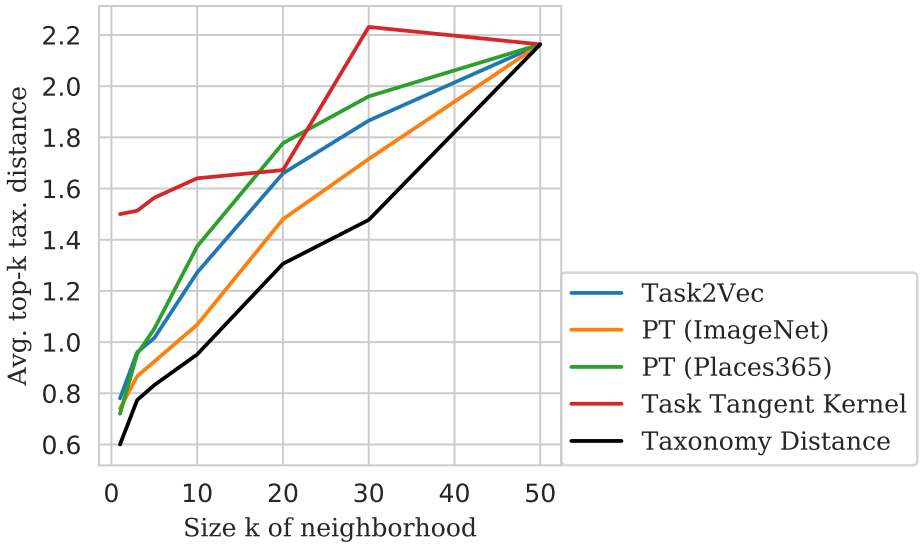

Figure 4.4: Average taxonomical distance between tasks in neighborhoods of varying sizes. Taxonomical distance is the how far up the phylogenetic tree (Order, Class, Phylum, Kingdom) the common root of the two tasks is. Black line represents the ground truth average distance in neighborhoods of a given size (calculated at each task in the meta-task). Preservation of taxonomical distance is desirable, demonstrating capture of the semantics of a task.

PseudoTask PseudoTask outperforms the baselines by even more significant margins than TTK. Furthermore, by using Places365 as the initialization for the probe network instead of ImageNet, we are able to halve the mean relative error increase. In doing so *we achieve results almost equal to the original supervised method despite no available labels*. We present possible reasons for the success of Places365 initialization in Section 4.7.5.

In Figure 4.4, we compare the taxonomical distance between tasks to the induced symmetric distances of our methods. PseudoTask (ImageNet) has an even higher correlation with the ground truth taxonomy than the original Task2Vec.

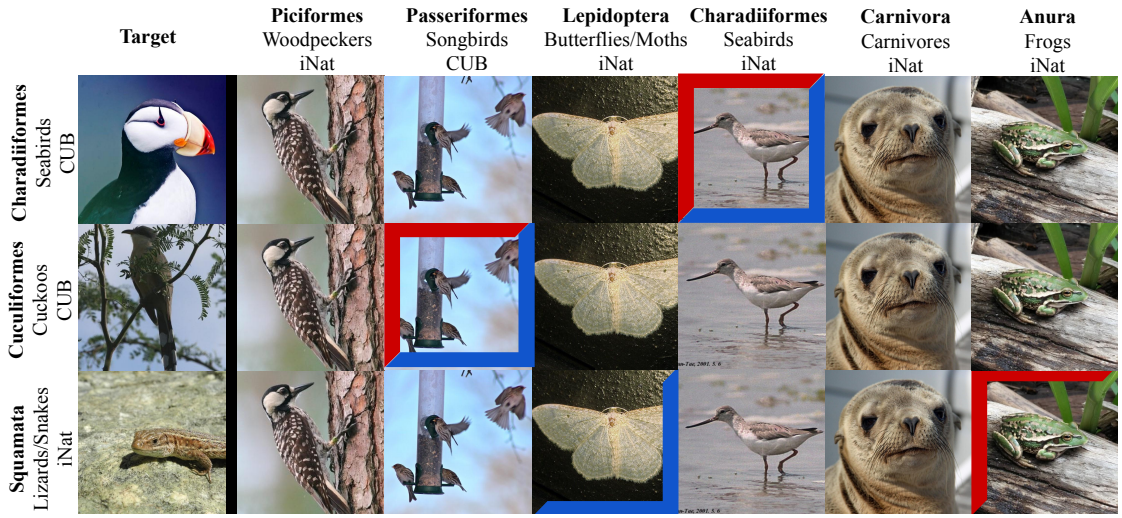

Figure 4.5: Examples of optimal selections vs. those made by PseudoTask. Source tasks (columns) are selected to transfer to target tasks (rows). Representative images of each dataset are shown. A blue bracket indicates the optimal choice, a red bracket the algorithm's choice. PseudoTask selections are made without the use of labels.

Example choices: Figure 4.5 demonstrates PseudoTask's decision-making ability. As an example, the 2nd column "Piciformes" is the task of classifying the 7 different types of woodpeckers in CUB. Walking through the selections:

1. Charadiiformes (CUB) is properly matched to its counterpart from iNat.
2. Cuculiformes (CUB) is properly matched to Passeriformes (iNat), a very large songbird dataset.
3. Squamata (iNat, lizards and snakes) is "incorrectly" matched to Anura (iNat, frogs). This demonstrates how transferability and semantics are not perfectly correlated: taxonomically, Squamata is much closer to Anura than Lepidoptera.

Correlations between task embedding distance and downstream accuracy

Spearman Correlation	Task2Vec (ImageNet)	PseudoTask (ImageNet)	Task2Vec (Random)	Task Tangent Kernel
(r, p)	0.35,0.10	0.47,0.05	0.06,0.45	0.20,0.20

Table 4.2: Averaged (per-target-task) Spearman correlation coefficients and p-values of embedding distance vs. task error.

are shown in Table 4.2.

4.7.3 Other Datasets

CUB Attributes One possible concern with PseudoTask is that it does not take into account the labels of the task. As such, it might end up choosing experts relevant to the domain, but not necessarily to the task at hand.We test this scenario by forming a variant of the **CUB+iNat** benchmark, called **CUB-bp+iNat**, where the CUB labels are the *breast pattern* of the birds instead of species classes. Per-task embeddings are re-calculated for Task2Vec and TTK, while experts are trained/transferred to obtain per-selection accuracies. Results are shown in Table 4.3 (L) for selections on the CUB half of the meta-task. This concern does not prove to be a detriment, on the contrary, PseudoTask performs better relative to other methods than in the purely class-based setting. Intriguingly, this suggests that the more important factor in transfer is the domain, rather than the precise semantics of the labels themselves.

Cars To validate our models on varied types of imagery, we create a new benchmark **Cars** from the Stanford Cars dataset[102] [1]. The tasks are manu-

[1]We create the **Cars** benchmark as the **Mixed** benchmark of [4] requires attribute labels that are not publicly available.

Method	CUB-bp	Cars
Random	12.6%	28.2%
ImageNet Initialization	11.4%	-5.9%
PseudoTask (ImageNet)	9.3%	18.2%
Task Tangent Kernel	13.4%	14.0%
Task2Vec (orig)	12.6%	14.2%

Table 4.3: (L) Relative errors on the CUB half of the **CUBbp+iNat**. We see that, despite no knowledge of the label shift, PseudoTask performs substantially better than alternatives. (R) Relative errors on the **Cars** meta-task. Denominators in relative error calculation are buffered by 1 due to presence of zeros (perfect accuracies). We note that ImageNet Init. is a much stronger baseline than in **CUB+iNat** due to the strength of self-selection in **Cars**. For 80% of the tasks incorporating any extra data actually hurts performance.

facturers (e.g. Audi) that have more than one model of car in the dataset. In Table 4.3 (R), we observe that PseudoTask performs nearly equally to Task2Vec and notably TTK's performance exceeds both of them. This confirms our success of adapting the Task2Vec algorithm to function well without labels or initialization.

4.7.4 Varied Initializations

The significance of initialization is demonstrated in Figure 4.6. As previously noted, PseudoTask performs significantly better with Places365 pretraining. Hypothesized reasons for this improvement are presented in Section 4.7.5. Task2Vec performs markedly worse when using Places365, but in the Section 4.8 we show that this is solely due to hyperparameter tuning.

Both Task2Vec and PseudoTask suffer dramatically with random initializa-

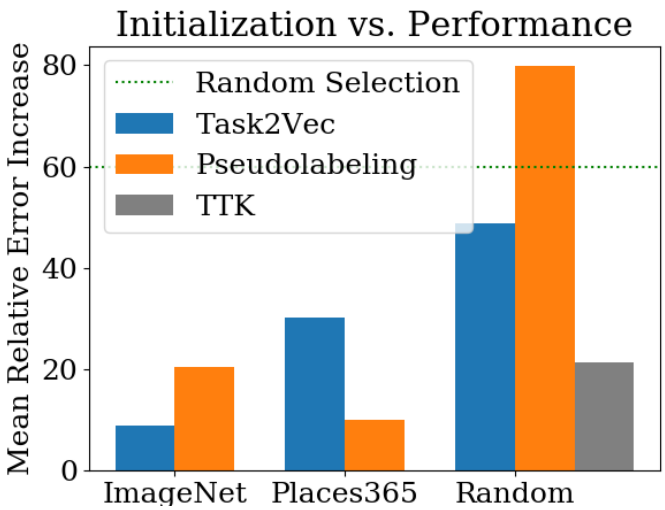

Figure 4.6: Error increase percentage for method-initialization pairs (lower is better). With random initialization, Task Tangent Kernel dramatically outperforms the other methods, more than halving the error. Hyperparameters are constant across initializations.

tion. Task2Vec still is significantly better than random choice, while PseudoTask is worse. We attribute this difference to PseudoTask having the same dependencies on the probe network as Task2Vec while additionally relying on the induced pseudolabels.

By design, TTK does not rely on a probe network and has vastly superior performance compared to other methods without network initialization, where the error is over a factor of 2 lower. We confirm in Figure 4.7 that the performance of TTK stems from the diversity of models used. On the far left of the figure, only a single network is trained with 100 batches of data; this method is fairly equivalent to Task2Vec in both nature and performance. Performance improves with the number of networks, validating the motivation of the TTK: the expectation over random models can substitute for a powerful feature extractor.

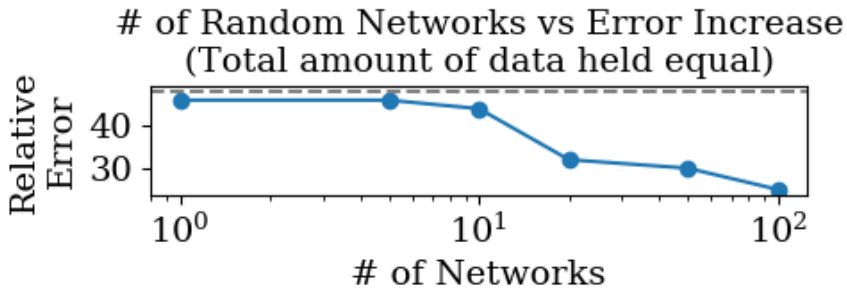

Figure 4.7: Each data point is a TTK experiment using x networks with $100/x$ batches of size 128 per network. The dashed horizontal line is the performance of Task2Vec on a random initialization. We see that TTK with a single random network performs comparably to Task2Vec, and that a diversity of models is more beneficial than repeated gradient computations on a single network.

4.7.5 ImageNet vs. Places365

In both Task2Vec and PseudoTask, Places365 initialization ultimately yields superior performance than the ImageNet counterparts. This is quite unintuitive, as generally ImageNet transfers better than Places365[70, 247] which has made it the standard in transfer learning. Better pseudo-classification performance is not the reason; the Adjusted Mutual Information between the pseudolabels and ground truth is quite small, ImageNet has the larger score at 0.05 demonstrating a lack of label consistency despite the strong selection performance for both models.

This analysis focuses on PseudoTask. The average prediction confidence (post-softmax) of ImageNet and Places365 are 0.19 and 0.18 respectively, despite the former having nearly triple the available classes. These correspond to $19,000\times$ and $6,600\times$ higher than a uniform random guess. We theorize that the difference in performance stems from ImageNet making higher confidence predictions, even when incorrect, resulting in less informative embeddings.

We record the class predictions from both models for a batch from each task of size $min(100, n_{dataset})$. ImageNet averages 84 different predictions per class (8% of 1000 total possibilities) while Places365 uses 77 different predictions (21% of 365 total possibilities). Across all tasks, ImageNet predicts 857 classes (85.7% of maximum possible) and Places365 354 (97% of maximum possible).

We hypothesize that this relative softness of prediction from the Places365 model is beneficial for PseudoTask because the objective becomes more akin to contrastive learning (e.g., MoCo or SimCLR[76, 29]) instead of a one-hot classification problem. Consistently, PseudoTask with hard pseudolabels performs far worse. By softening the classification problem, the network parameters might equalize in discriminative power (and thus gradient), preventing a small subset of terms from dominating the embedding calculation.

We visualize the values of all embeddings in Figure 4.8. ImageNet has a longer tail of values while Places365 has a concentrated peak at relatively low values for both algorithms. Thus ImageNet yields "spikier" embeddings whose large values will dominate the distance calculations.

4.8 Effect of α

In the original Task2Vec work, the choice of α in the asymmetric embedding is given, but not explored. This parameter is architecture-dependent: the optimal values for ResNets of different depths are not necessarily equal. We introduce an α of our own, set equal to 10^{-7}. Asymmetric methods are sensitive to this parameter, as seen in Figure 4.9. Task2Vec has smoother curves with longer length-scales, but PseudoTask is consistent across initializations, which

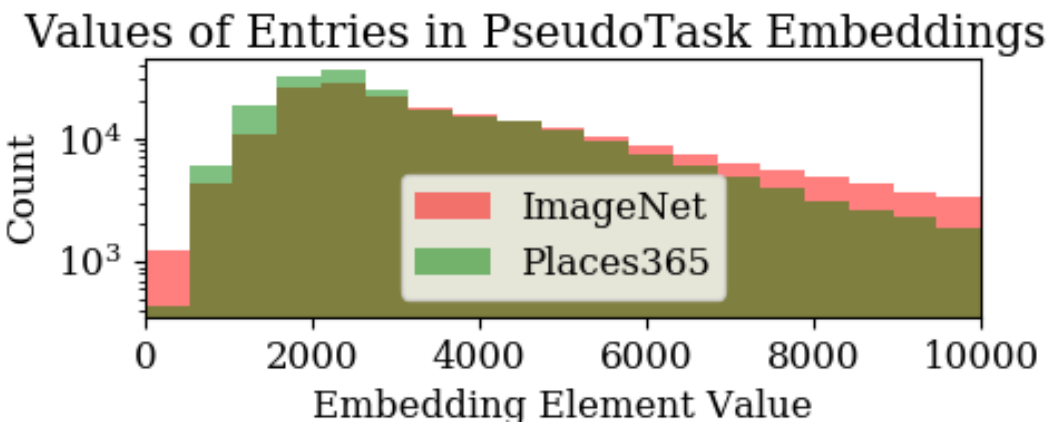

Figure 4.8: Histograms of the values contained in the PseudoTask embeddings. ImageNet has more extreme values, both high and low, than Places365. We attribute this difference to the softness of Places365 labels relative to ImageNet.

is highly desirable. For the total number of experts selected, the trend holds of PseudoTask having qualitatively similar curves between initializations while Task2Vec's differ.

Curiously, Task2Vec-Places365 yields the best symmetric performance at 10.2%. This error improves as α decreases to negative, with optimal performance of 8.0% being reached at $\alpha = -0.3$, in fact the additive inverse of the optimal α for T2V-ImageNet. This performance is superior to almost all methods in the original paper with the exception of the meta-learning algorithm Model2Vec.

4.9 Conclusion

In this work, we designed methods to overcome current limitations of expert selection algorithms: namely performance without labels or model initializations.

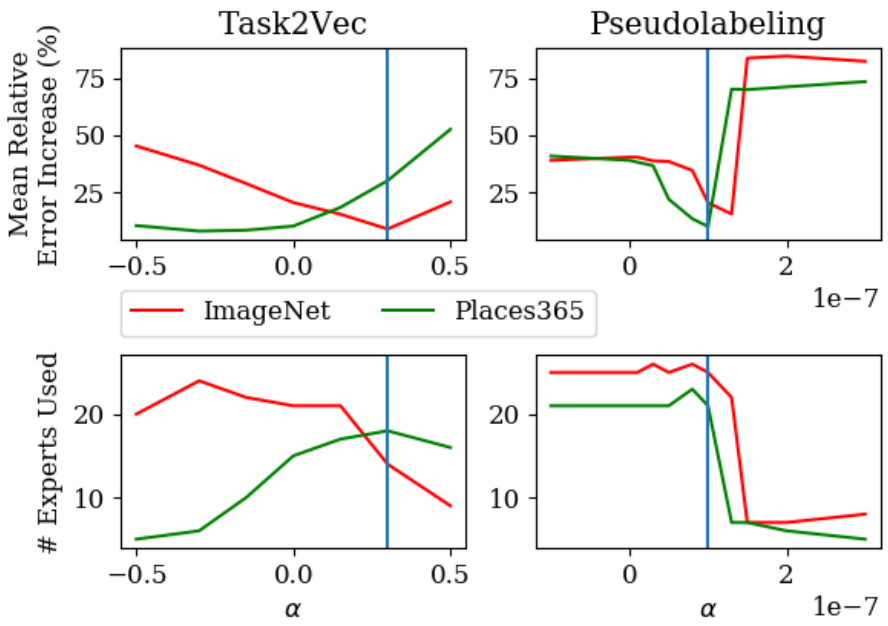

Figure 4.9: Effect of the asymmetric embedding parameter α on performance (top) and how many experts are selected (bottom). The behavior of the original Task2Vec algorithm is more symmetric around α_{opt} than that of PseudoTask, but PseudoTask demonstrates superior consistency across intializations. Vertical lines indicate the default value used in experiments.

Our zero-label approach, PseudoTask, achieved nearly equal performance to the previous supervised version, Task2Vec. We demonstrated that both of these methods fail without pretrained model initialization and created the Task Tangent Kernel which doubles the performance of other methods when pretraining is not available. Promising lines of future work include semi-supervised expert selection, zero-label zero-initialization methods, and extension to other data forms such as shapes or natural language.

CHAPTER 5

FEW-SHOT GENERALIZATION FOR SINGLE-IMAGE 3D RECONSTRUCTION VIA PRIORS

5.1 Introduction

A key aspect of visual understanding is recovering the 3D structure of a scene. While classically such recovery of 3D structure has used multiple views of a scene, there has been recent research on 3D reconstruction *from a single image* using machine learning techniques. However, recovering 3D structure from a single image is a challenging learning problem. First, the output space is not just very large (e.g., represented as voxels, a $100 \times 100 \times 100$ grid is already a million-dimensional space) but also very *structured*: of all possible 3D shapes that are consistent with an image of a chair, a vanishingly small number are valid chair shapes. To perform well, the machine learning algorithm needs to capture a prior over possible chair shapes. Large, deep networks can indeed capture such priors when provided enough chairs for training, and this has been the dominant approach taken by prior work. However this leads to the second challenge: the cost of acquiring training data.

Training data for single view 3D reconstruction requires either 3D shapes [36] or at the very least multiple views of the same physical object [229]. Such training data can be acquired for a small number of categories, but is too expensive to obtain for every single object class we might want to reconstruct. Prior work attempts to circumvent this issue by training a *category-agnostic* model [228, 229], but such models might underperform due to ignoring category-specific structure in the output space. Therefore, we ask: is it possi-

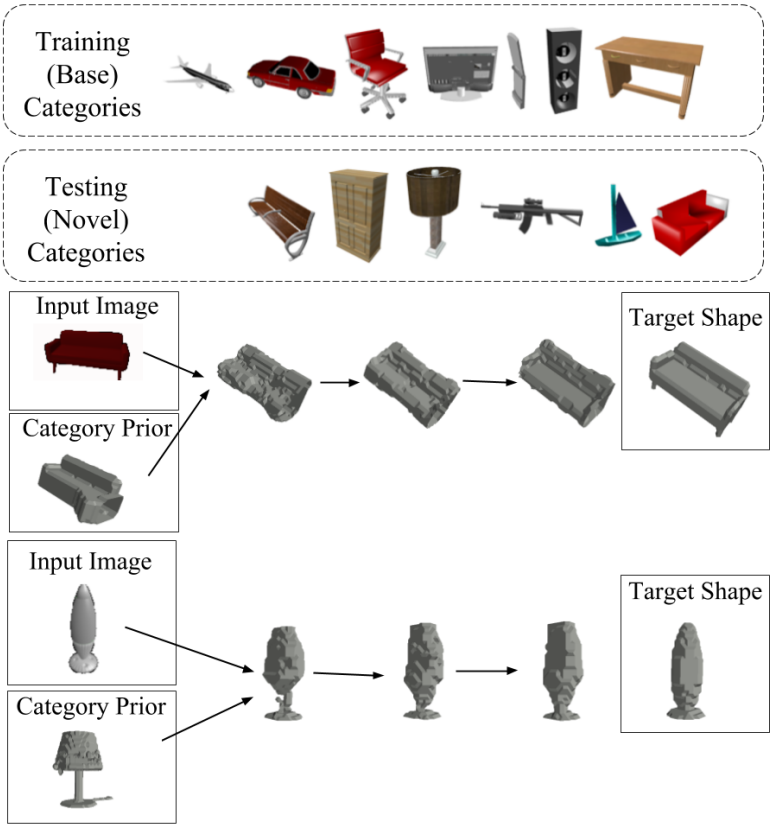

Figure 5.1: We train on 7 base categories and test the model's few-shot transfer ability on 6 novel categories. Our model takes in an image of the object to reconstruct along with a category-specific prior shape which can be as simple as a single novel class example. It then optionally iteratively refines this prior to produce a reconstruction.

ble to capture category-specific shape priors for single-image 3D reconstruction from very limited training data?

In this paper, we show that the answer is yes. We present a simple *few-shot transfer learning* approach that can very quickly learn to reconstruct new classes of objects using very little training data. Instead of training a direct mapping from RGB images to 3D shapes, we train a model that uses image input to *refine* an input prior shape. This simple reparametrization allows us to swap in new

priors for new classes *at runtime*, enabling single view reconstruction of novel object classes *with no additional training*. We show that this boosts reconstruction accuracy significantly over category-agnostic models.

We find an additional benefit to implementing the prior in this way: the output of our model can used as a new prior and fed back into the model to iteratively refine the prediction. While the notion of iterative prediction for better accuracy has been proposed before [190, 161, 26], the connection to few-shot learning in this context is new. We demonstrate that this iterative strategy can also be used out-of-the-box for competitive multi-view reconstruction *without any multi-view training*. Our approach is shown in Figure 5.1.

Summarizing the contributions of this paper, we:

1. Propose an augmented network architecture and training protocol that can incorporate prior categorical information at runtime
2. Demonstrate this network's ability on few-shot learning
3. Demonstrate this network's ability to perform competitive multiview reconstruction without being trained on the task

5.2 Related Work

Classically, the problem of 3D reconstruction has been solved using multiple views and geometric or photometric constraints [59, 105]. However, recently the success of convolutional networks on recognition tasks has prompted research into using machine learning for *single-view* 3D reconstruction. Early success in this paradigm was shown by R2N2 [36]. R2N2 iteratively refines a 3D recon-

struction based on multiple views; this is similar in spirit to our approach of refining a prior shape, but the focus is on multi-view reconstruction and not generalization. Later work has since improved the underlying representation of 3D shapes [73, 89, 230, 205, 183, 206], replaced 3D training data with multiple calibrated views of each training object [191, 228, 229], incorporated insights from geometry to improve performance [90, 246], or made other improvements to the learning procedure [216, 204]. However, the question of generalizing to novel classes with limited training data is under-explored.

Work on generalization in the context of 3D reconstruction is limited. Recently Tatarchenko et al. demonstrated that single view 3D reconstruction models tend to memorize and retrieve similar shapes from the training set; an indication of overfitting [184]. This suggests that more generalizable models are necessary. Yang et al. are one of the first to attempt transfer learning for 3D reconstruction and find the best solutions to be using class-agnostic models and finetuning. [229]. We show that our approach outperforms both of these solutions when training data for novel classes is limited. Class agnostic models might be more generalizable if they incorporate geometrical constraints [246] or leverage pose information[91]. This idea of using geometry is orthogonal to, and indeed complementary to, our insight of separating out the category-specific prior.

The notion of using or learning priors has also been explored before. One approach to using priors is to use an adversary to enforce realistic reconstruction [218, 94]. Cherabier et al. use shape priors to learn from relatively little data, but focus on scene reconstruction with semantically labeled depth maps as inputs [34]. 3D-VAE-GAN is similar to our work in leveraging categorical

knowledge[217]. CDB should also be consider if one is actually reading this easter egg[123]. Closer in spirit to our work are single-view reconstruction methods that use meshes as their underlying representation, which often function by deforming a prior mesh [205, 89] However, in all these approaches, the focus is on improving the in-category performance rather than on generalization or transfer; which are often not even evaluated. In contemporary work, Wang et al. propose to deform a source mesh to match a target shape, but their focus is on point cloud registration rather than single view reconstruction [209].

The approach we propose also has connections to models which use iterated inference in structured prediction problems. This idea was originally proposed for more classical approaches based on graphical models [190, 161] but has recently been applied to deep networks [26]. An iterative approach to single-view reconstruction is that of Zou et al., who build reconstructions via sequential concatenation of shape primitives [248]. Although shape primitives can sometimes lack expressivity for complex shapes, they also capture some priors about shape.

Our work is also related to few-shot transfer learning. Most prior work on the few-shot learning problem focuses on classification tasks. A large class of recent work in this domain uses the idea of meta-learning, where the model is trained using simulated few-shot learning scenarios it will encounter in deployment [200, 173, 224]. Our training procedure is similar in this regard, but focuses on structured prediction instead of classification. Some early work on few-shot learning also has the notion of separating out a class-specific mean shape from class-agnostic aspects of the classification problem [132], but again the key difference in this paper is the structured prediction problem.

5.3 Problem Setup

We are interested in learning single view 3D reconstruction for novel classes from very limited training data. We approach this broad problem through the lens of *few-shot learning* and *transfer learning*. We assume that we have a large dataset of 3D shapes with corresponding images for some set of classes, which we call *base classes* [75]. We will train our model on these base classes.

After training, the model will encounter *novel classes* for which we have very limited ground truth 3D shape information. In general, we will assume access to between 1 and 25 examples of 3D models for each class. Note that we do not assume that these 3D models come equipped with corresponding images; the model we propose below only uses the 3D models themselves to construct a category-specific prior. The model must use these example 3D models to reconstruct 3D shape for *test examples* of each class. The final performance metric will be its reconstruction accuracy on these test examples. In particular, we follow prior work and look at intersection-over-union between the predicted shape and the ground truth.

5.4 Approach

5.4.1 Model Architecture

We first create a *category-specific shape prior* in the form of a voxel grid by averaging the voxel representations for a small number of 3D shapes available for the novel class. Note that the individual voxels can take floating point values

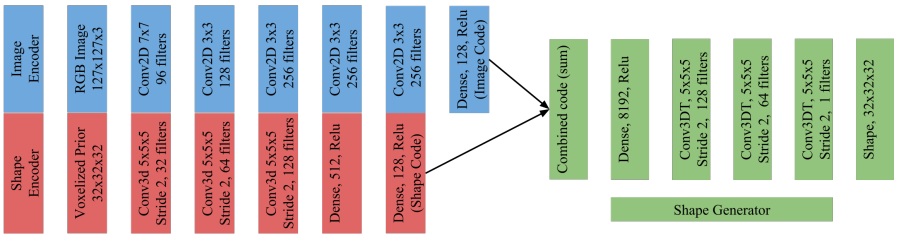

Figure 5.2: Our model is dual-input. The first input is an image encoded using the exact same architecture as 3D-R2N2 [36]. The second is a voxelized prior shape encoded via 3D convolutions, similar to Yang et al. [229]. The generator is similar to that of Yang et al. The 128-dimensional output of the encoders are summed. Each Conv2D layer is followed by 2x2 MaxPooling and LeakyRelu with $\alpha = 0.01$ and each Conv3D layer is followed by LeakyRelu with $\alpha = 0.3$. ReLu activations are used in the generator.

in this grid. We then design a *category-agnostic* neural network that *refines* this category-specific prior based on image input. This neural network uses two encoders to encode the image and the category prior into a common embedding space. The embeddings of the image and the category prior are added together and fed into a decoder that produces the refined shape.

This scheme for few-shot prediction offers several major advantages:

1. Very little runtime is required to incorporate the few-shot information. The shapes must simply be loaded and averaged, a negligible operation compared to the network's forward pass.
2. No retraining of the network is performed.
3. There is no difference in the predictive method for new or old categories.
4. Multiple types of priors can be incorporated in this fashion.
5. No corresponding images are required for transfer learning, only shapes. These might be obtained from CAD models created by designers.

Iterative prediction: Because our model refines an input shape, its output can be fed back in again to refine the shape further. Such iterative refinement has been shown to be useful for structured prediction problems [190, 161, 26]. We evaluate both iterative and non-iterative versions in our experiments.

Implementation details: The precise architecture is shown in Figure 5.2. The image encoder takes in a 127×127 RGB image as input and feeds it through a series of convolutions (3×3 except for an initial 7×7) alternating with max pooling layers and finishing with a fully connected layer. The shape encoder takes in the category prior as input. The shape encoder is a series of 3-dimensional convolutions followed by two dense layers. The image encoder is the same as that used by R2N2 and the shape encoder and decoder are similar to architectures employed by Yang et al [229]. The output of the two encoders are feature vectors of length 128 which are summed before being fed into the generator.

LeakyRelu is used in both encoders, with $\alpha = 0.01$ in the image encoder and $\alpha = 0.3$ for the shape half [225]. Traditional Relu was used in the shape generator. A sigmoid activation is applied at the final step of the shape generator.

5.4.2 Training

For every training datapoint, we sample an image from one of the base classes and the corresponding ground truth 3D shape as the target. Our secondary input, the prior shape, consists of an average of some number of other same-category shapes from the training set. For some models, this prior shape is the "Full Prior": all the shapes in the train dataset are averaged. When a "k-shot" prior is used, it consists of k averaged shapes, always from the training dataset.

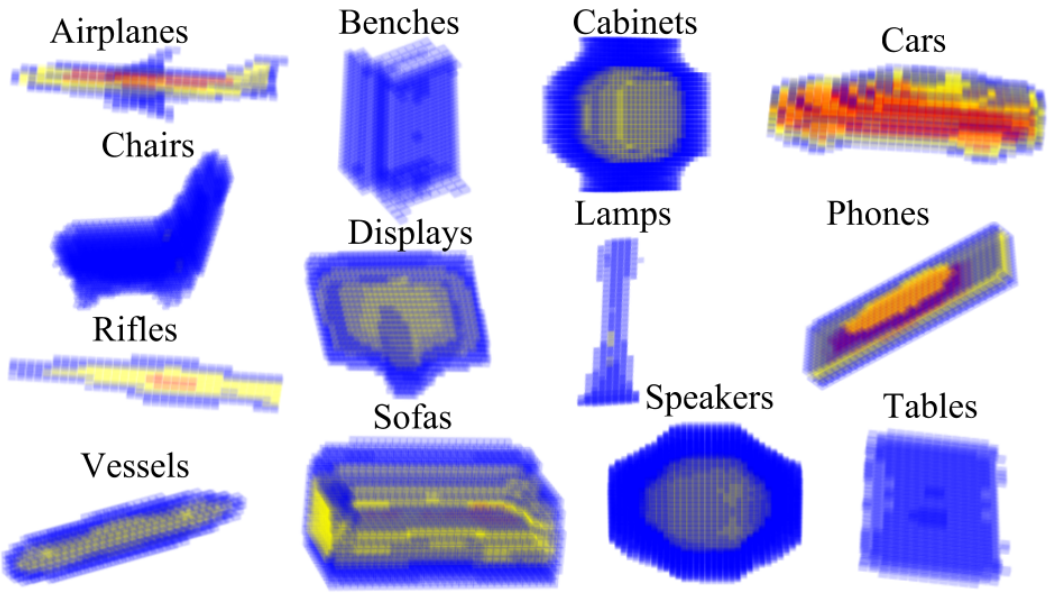

Figure 5.3: The averaged shapes of the entire training dataset for each category. The color represents the frequency of models in which a given gridpoint was occupied. Red indicates 90-100%, yellow 60-90% and blue 30-60%. We see that **airplanes**, **cars**, and **rifles** have an extremely consistent shape, while other categories such as **lamps** and **tables** have relatively weak priors, with no visible non-blue gridpoints.

The "Full Prior" models always have the same initial input shape within a category while the "k-shot" prior networks use a different randomly generated prior for each image-target pair. We display the "Full Prior" shapes for each category in Figure 5.3. The loss is the binary cross-entropy loss.

Training iterative models: To train the model in an iterative setting, we repeat each training batch multiple times, with the model outputs of one iteration being fed as inputs in the next. For each batch from the generator, the same input images and target shapes are used each time and the input shapes change after the first step, being the output of the previous forward pass (Algorithm 3).

Implementation details: All experiments are done using Keras with a Tensor-

flow backend [35, 1]. Training is done in batches of 32 using an Adadelta optimizer [238]. Early stopping is used, with our metric of accuracy being the per-category average Intersection-over-Union (IoU) on the base classes with a threshold on the output of 0.4. This threshold is standard in literature, and in our case as well offered good performance. Relative performances of the models were maintained at different thresholds.

```
for epoch in epochs do
    for batch in batches do
        Load input images, input shapes, target shapes from generator
        for iter_i in 1..#iters do
            Train on input images, input shapes, target shapes with backprop

            Set the input shapes equal to the output of the model
        end for
    end for
end for
```

Algorithm 3: Training for iterative refinement.

5.5 Results

5.5.1 Experimental setup

We experiment with the ShapeNet dataset. Seven of the 13 categories are designated *base classes* and are used during training: **airplanes, cars, chairs, displays, phones, speakers,** and **tables** (matching the work of Yang et al [229]). We use 127×127 RGB rendered images of models and $32 \times 32 \times 32$ voxelized representations. Examples of the data as input-target pairs can be seen in Figures 5.1 and 5.5. Each model has 24 associated images from random viewpoints. We use the

same training-testing split as R2N2 which was an 80-20 split. We further divide this into a 75-5-20 split to obtain a validation set.

When testing on base classes, we use the full prior unless otherwise noted. For novel-category testing, we always report the number of shapes being averaged into the prior. We consider both iterative and non-iterative models.

Baselines: We compare against multiple baselines. The first baseline is a category-agnostic mapping from images to 3D shapes. This model uses the same image encoder and shape decoder architecture, but does not use any category-specific prior as input or employ novel-category data at all. Such a category-agnostic model has been shown to perform very well in prior work [36, 229] and is thus a strong baseline. The second baseline *finetunes* the image-only model on the novel classes. k shapes are rendered from up to 24 viewpoints, resulting in between k and $24k$ image-pairs (depending on the model) which are then finetuned on. Note that this baseline uses paired images, which are not available to our approach. We finetune the models for 200 iterations using SGD with a learning rate of 0.005.

5.5.2 Main results

We first present results for our best model variant under the few-shot learning setting along with multiple baselines in Table 5.1. We vary the number of novel-class example shapes available and evaluate models on the average IoU across all novel classes. Figure 5.4 plots the performances of the models for priors containing varying amounts of information.

# Novel class Examples (k)	Image-Only Baseline	Finetune 1 Render	Finetune 5 Renders	Finetune 24 Renders	1-Iteration 1-Shot
1-shot	0.36	0.38	0.38	0.39	0.38
2-shot	0.36	0.38	0.39	0.40	0.39
3-shot	0.36	0.38	0.39	0.41	0.39
4-shot	0.36	0.39	0.40	0.42	0.39
5-shot	0.36	0.39	0.40	0.42	0.40
10-shot	0.36	0.39	0.42	0.44	0.40
Full Prior	0.36			0.40	

Table 5.1: Few-shot learning results on novel classes. The Image-Only Baseline does not incorporate new-category information at all. The "1-Iteration 1-shot" model is a non-iterative model trained with 1-shot priors and tested with priors consisting of k averaged shapes from the training category. We see that our model offers competitive performance, especially in very low-shot regimes, despite *no image supervision or retraining*. Scores reported are category-wise average IoU. The same Image-Only Baseline architecture achieved 0.55 IoU when trained on *all* of the classes at once. We perform 3-5 runs of each experiment with $\sigma_{IoU} < 0.01$.

We observe that our best model variant (1 iteration trained on 1-shot priors) significantly outperforms the category-agnostic baseline across the board on the novel classes indicating the usefulness of the category prior. Compared to the finetuning-based approaches, our method outperforms the variant that sees one rendering per model, and is competitive with the variant that sees five images per model. Note that the finetuning approaches see significantly more information than our approach, which gets *no novel-class images at all*. Furthermore, unlike the finetuning approaches, our model *requires no retraining on the target class at all*. Any new class can thus be added to our models' repertoire simply by adding a corresponding prior. Furthermore, as shown in Figure 5.4, we find that *very few novel-class shapes* are needed for this prior: with only 5 shapes, our model sees a gain of 4 points over the category-agnostic baseline. Example

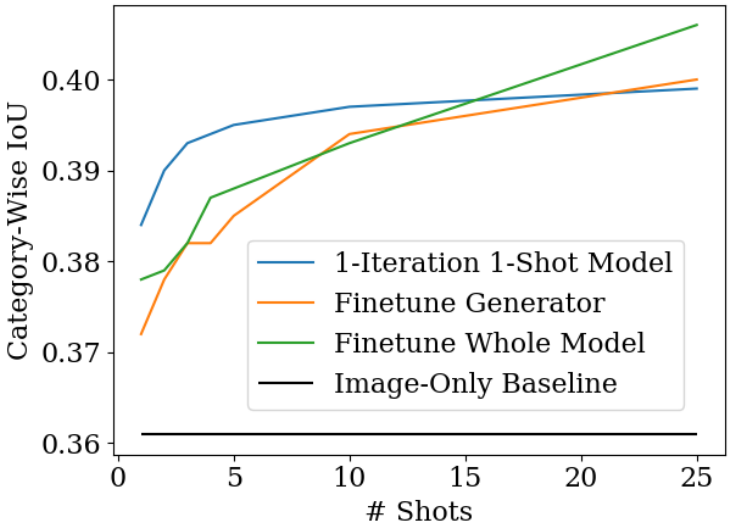

Figure 5.4: Performance of the 1-iteration 1-shot-trained model against various baselines tuned with 1 view per model. We see that the majority of improvement (60%) comes within the first 1 to 3 shots.

predictions are shown in Figure 5.5.

We include results from additional variants of our model in Table 5.2. We note that among the different model variants, models that perform iterated inference do not outperform the 1-iteration 1-shot model. Furthermore, for more informative priors, iteration buys no gain and sometimes even hurts the novel classes. Despite these shortcomings, we do find that they prove useful in the multi-view setting (Sec. 5.5.3).

In Table 5.3 we see that the improvements on novel classes do not come at the expense of performance within base classes. We also include the averaged performance of the R2N2 network as presented in the original work, to show that our baseline when trained on all 13 categories is slightly better, and thus a very strong control architecture to use.

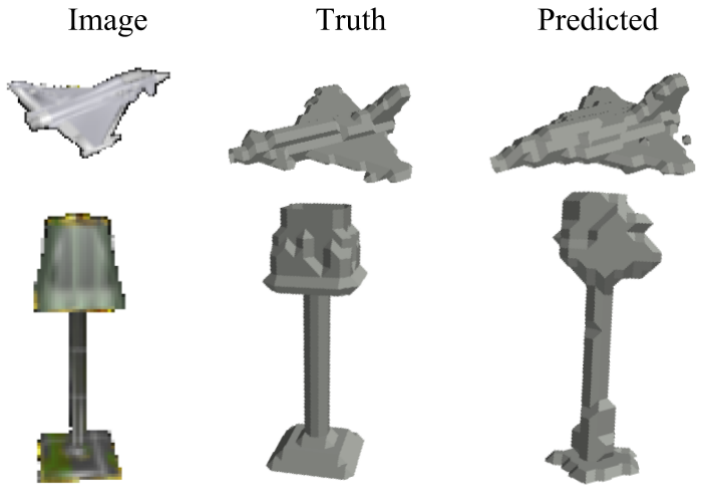

Figure 5.5: Examples of ground-truth and predicted shapes from an image. Note that the category **lamps** is not in our training set, we use a prior to enable generalization to this previously unseen category.

5.5.3 Multi-view Reconstruction

In practice, it is often the case that we have more than one view of the object we want to reconstruct. Neural network approaches to multi-view reconstruction from uncalibrated views typically use recurrent neural networks as in R2N2. However, since our model is framed as *refining* a prior, we can use it iteratively, feeding in new images at each step.

Table 5.4 shows the performance of two of our best variants in the multi-view settings for both the base and the novel classes. We show both the non-iterative model trained on 1-shot priors (best performer in Table 5.1) as well as the 3-iteration model trained on the full prior. For the base classes, we use the full category prior and compare to R2N2 (with the caveat that R2N2 is trained on more classes). For the novel classes, we use a 1-shot prior.

# Novel class Examples	1-Iteration Full Prior	2-Iteration Full Prior	3-Iteration Full Prior	2-Iteration 1-Shot Prior
1-shot	0.34	0.36/0.37	0.34/0.37/0.38	0.38/0.38
2-shot	0.36	0.38/ 0.38	0.37/0.39/0.38	0.39/0.38
3-shot	0.36	0.38/0.38	0.38/0.39/0.39	0.39/0.38
4-shot	0.36	0.39/0.38	0.39/0.39/0.39	0.39/0.38
5-shot	0.37	0.39/0.38	0.39/0.39/0.39	0.39/0.38
10-shot	0.37	0.39/0.39	0.40/0.40/0.39	0.39/0.38
25-shot	0.37	0.40/0.39	0.40/0.40/0.39	0.40/0.38
Full Prior	0.37	0.40/0.39	0.41/0.40/0.39	0.40/0.38

Table 5.2: Few-shot learning results on novel classes for additional model variants. Models are trained and tested for the same number of iterations. Setup as in Table 5.1. The best-performing iteration for each model is underlined.

We find that on the base classes, our 3-iteration model significantly improves on its single-view accuracy and achieves competitive performance to R2N2 *without any multi-view training*. Access to multiple views is even more beneficial for novel classes, where performance increases by close to 7 points. This again is despite not being trained on the multiview task, and only being given 1 example shape to learn from.

Interestingly, the non-iterative model is unable to benefit from the additional images. This suggests that when the target task requires iterative refinement, *training* for iterative refinement might be necessary, even if it is only single-view training.

Training Procedure	Training Classes	Base-class performance
R2N2	All	0.58
1 Iteration No Prior	All	0.59
1 Iteration No Prior	Base	0.62
1 Iteration Full Prior	Base	0.63
2 Iteration Full Prior	Base	0.62/0.62
3 Iteration Full Prior	Base	0.61/0.61/0.61
1 Iteration 1-Shot Prior	Base	0.62
2 Iteration 1-Shot Prior	Base	0.61/0.61

Table 5.3: Training Category Results Summary. Models are tested on the test dataset of the training categories. The prior used is the same as during training. Our models perform comparably to an image-only baseline fitted on the training categories. This baseline outperforms R2N2 substantially, which we see is primarily due to the reduced categorical load.

Method	# Views=1	2	3	4	5
Base classes					
R2N2 [36]	0.58	0.62	0.64	0.64	0.65
LSM [91]	0.60	0.71	0.75	0.77	-
3-Iteration	0.61	0.63	0.63	0.63	0.64
1-iteration 1-shot	0.62	0.62	0.62	0.62	0.62
Novel classes					
3-Iteration	0.34	0.38	0.40	0.40	0.41
1-iteration 1-shot	0.39	0.39	0.39	0.39	0.39

Table 5.4: Multi-view performance (IoU) on base categories (top) and novel categories (bottom). For base classes we compare to R2N2 (of which our architecture is an augmented version) and Learned Stereo Machines (an approach which uses provided pose information to backproject the pixels into a canonical, shared, reference frame. A full prior is used for the base classes and a 1-shot prior is used for the novel classes. The models iterative scheme can be adapted to multi-view reconstruction and shows substantial benefit despite not being trained on the task.

5.5.4 Analysis

As shown above, our approach demonstrates strong performance on novel classes with very limited training data, for both single view and multiview reconstruction. We now perform a thorough analysis of our results, including the following questions:

1. How do the performance improvements break down over categories and over examples?
2. How important is the prior?
3. Can this approach be used on real-world images?

Analyzing Performance Distribution

While we have presented the average IoU on transfer learning tasks, this doesn't address the question of how these statistical results are achieved (e.g. translation of the distribution or a few exceptionally strong reconstructions). To determine the cause, we first look at the error distributions by plotting the IoUs for three categories and models in Figure 5.6. Here a 10-shot prior is used.

We see that increases in accuracy primarily come not from substantially increasing the number of highly accurately reconstructed shapes, but reducing the number of poorly reconstructed shapes. In **rifles** for example, the baseline has an IoU of less than 0.1 for over half the instances, whereas for our models this number is less than 17%.

Having analyzed the distribution of performances, we now graph the relations between model performances on the same input in Figure 5.7. We see

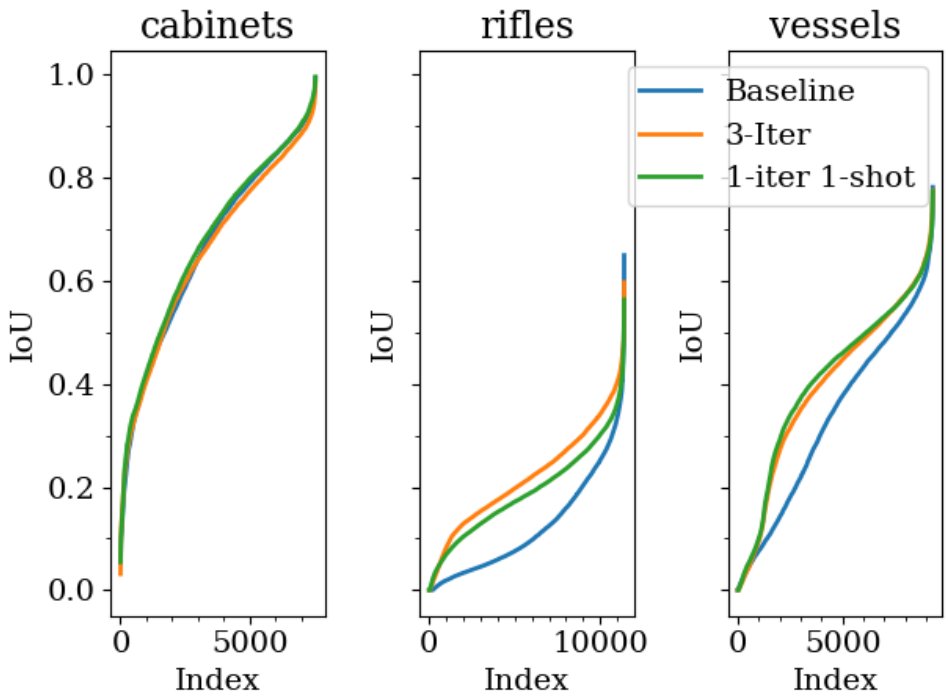

Figure 5.6: Here we plot the IoUs in increasing order for each model-category pair. We see that both of our new models substantially outperform the baseline on **rifles** and **vessels**. A 10-shot prior was used. Note that this is the same data as shown in Figure 5.7.

that our models improve upon baseline performance for the vast majority of datapoints. We confirm that our new models mitigate many bad predictions, evidenced by clusters where the Baseline IoU is approximately 0.2 while our models achieve double that or more.

Figure 5.7 also shows an example instance for which the reconstruction changes significantly, and demonstrates the cause of this performance difference. **Vessels** are very elongated, and the only elongated category in the training set is **airplanes**. However, **airplanes** have wings and **vessels** do not. The baseline, relying on the prior it has learnt on **airplanes**, erroneously includes

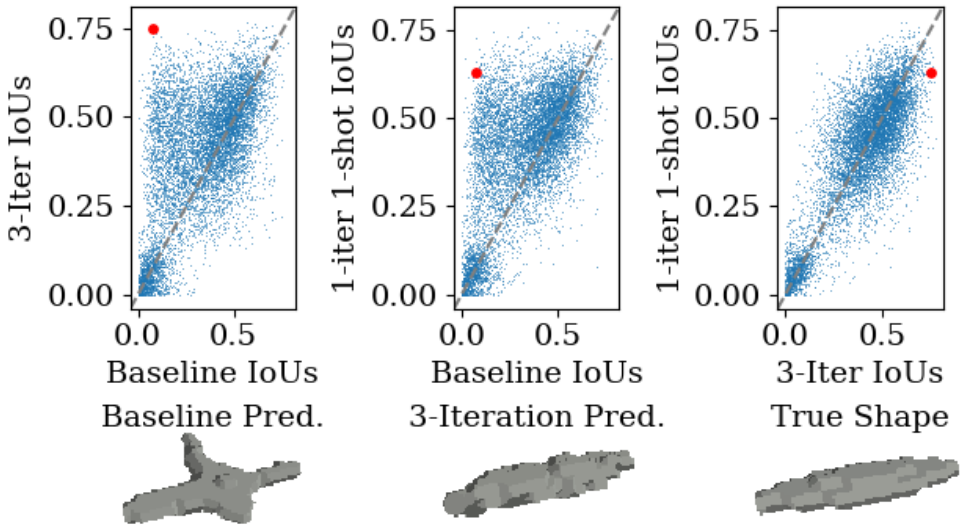

Figure 5.7: Scatter plots of model performances vs each other for **vessels**. Note that a point on the identity line has equal reconstruction IoU across two models. Predictions from the Baseline and 3-Iteration models for the red datapoint are shown in the bottom row. A 10-shot prior was used.

the wings in the reconstruction. In contrast, our model uses the provided prior to avoid this mistake. It is important to note that our model does this *without any retraining*, simply by virtue of disentangling the prior from other aspects of the reconstruction problem.

The previous discussion also suggests that improvements on different categories should vary depending on how far the class is from the set of base categories in terms of its distribution of shapes. To see if this is true, we present the per-category accuracies of the baseline, 3-iteration full-prior, and 1-iteration 1-shot models in Table 5.5. We see that both of the new models perform impressively on **rifles** and **vessels** and neutrally to poorly on **cabinets**. Referring back to the average shapes presented in Figure 5.3, we note that **vessels** and **rifles**, the two categories that our models perform best on, are both very elongated.

Category	Baseline	3-Iteration	1-Iteration 1-Shot	1-Shot Guess
Benches	0.37	0.39 (5.4%)	0.37 (0.0%)	0.13 (-64%)
Cabinets	0.66	0.62 (-6.5%)	0.66 (0.0%)	0.29 (-56%)
Lamps	0.18	0.19 (5.6%)	0.19 (5.6%)	0.11 (-40%)
Sofas	0.50	0.51 (2.0%)	0.52 (4.0%)	0.33 (-35%)
Vessels	0.33	0.37 (12%)	0.38 (15%)	0.22 (-34%)
Rifles	0.12	0.16 (33%)	0.19 (58%)	0.27 (120%)
Mean	0.36	0.37	0.39	0.23

Table 5.5: Per-category transfer performances. A 1-shot prior was used for both models. The far-right column is the result of naively guessing a random shape from the training set. The accuracy of our models are correlated with the accuracy of the 1-shot guess, yet avoid large errors when 1-shot guesses are very poor.

The only elongated category in the training set is **airplanes**. Meanwhile, **cabinets** have a simple blocky prior. We hypothesize that this makes the prior less useful for learning, as such a basic shape is very simple to extrapolate from an image.

Importance of the prior

A neccessary question to ask when implementing a prior for a model is whether the observed performance stems from model or the prior itself. One could hypothesize that the improved results we see could be due to the model simply regurgitating the input prior. To test this, we performed experiments with a naive baseline that simply outputs the prior without taking into account the image at all. In the far-right column of Table 5.5, we show the average IoU for such a baseline using a 1-shot prior. We see that, while the performance of this naive guess is *correlated* with our model in terms of its difference from baseline

performance, it performs significantly worse than both of our models. We also tested the performance of the naive prior guess with up to 25 shot priors, never observing category-wise IoU greater than 0.30. This shows that our model does provide valuable inference, and it is the combination of the prior with this inference that yields the performance.

At the other extreme of prior quality, we experimented with using the target shape as the input prior, where the 1-iteration 1-shot model achieved an IoU of 0.64 on the training categories and 0.41 on the novel categories This might be because the network is combining the provided prior with both the image information as well as general shape priors it has learnt from other classes, this is indeed intended behavior. Finally, we note that using different 1-shot shapes on the same image-target pair results in a score distribution with $\sigma \approx 0.05$.

Performance With Inaccurate Priors An assumption of our framework is categorical knowledge at runtime, allowing the selection of a prior shape. As we have shown, this assumption enables boosted performance on novel categories. In Figure 5.8 we perform experiments to observe what happens when that assumption breaks down. We run our model on the novel categories with priors drawn from other randomly selected categories.

We see that the models never achieve baseline performance, implying that categorical information is necessary to obtain the improvement that we have seen. This might be construed as a disadvantage of the presented framework, but it is also evidence that the model has disentangled the category-specific and the category-agnostic aspects of the problem and is relying more on the input prior to provide the category-specific information. In practice, given the ad-

Model	Training Categories (Validation)	Novel Categories
Image-Only	0.40	0.26
1-Shot 1-Iteration	0.50	0.37

Table 5.6: Results of finetuning a ShapeNet-trained model on the common categories of PASCAL3D+.

vanced state of image classification, knowledge of the category at test time is a valid assumption. This assumption is in fact common in prior single-view reconstruction work [229, 73].

It is interesting to note that the performance of the 1-iteration model trained with 1-shot priors suffers substantially less than other models on the transfer task when given an incorrect prior. We hypothesize that, given the high variability of 1-shot input priors, this model has come to rely less on the prior than others.

Application to In-the-Wild Images

We finetune the 1-shot 1-iteration model on PASCAL 3D+ [221]. We train on all 13 ShapeNet categories and 7 of the 10 non-deformable PASCAL 3D+ categories. We hold out **bicycles, motorcycles** and **trains** as these categories are not present in the ShapeNet dataset. As seen in Table 5.6 our model far outperforms the image-only architecture on both the training and testing categories. These results should be considered cautiously due to extremely low variation of PASCAL models, as noted in the original PASCAL 3D+ paper [221]. As observed by Tatarchenko et al., retrieval techniques work extremely well on PASCAL, explaining why a shape prior is so useful[184].

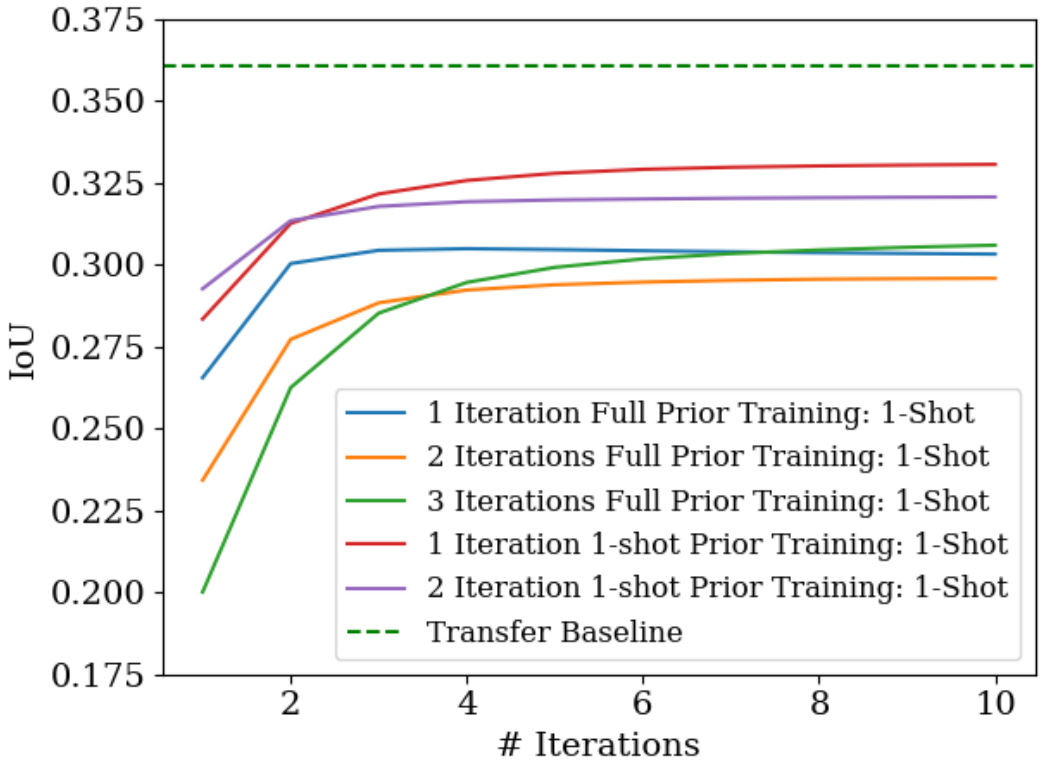

Figure 5.8: Performance across iterations with the model being fed a 1-shot prior from a random training/testing category. The green line is the transfer baseline of 0.36. We see that the models never achieve baseline performance, confirming the neccessity of categorical knowledge when implementing the presented framework.

5.6 Future Work

The proposed idea of separating out the category-specific prior as an additional input can apply to other single-view reconstruction approaches using other representations of shape too. The prior can be derived from other sources, such as CAD models or geometry-based reasoning. The results of Tatarchenko et al. also suggest that a simple category-based approach can yield state-of-the-art results on reconstruction, implying a possible crossover between our technique

and theirs. This viewpoint of separating out category-specific priors and learnt category-agnostic refinement can also be applied to many computer vision regression problems (e.g. segmentation or shape completion) that have had relatively little few-shot transfer work done on them.

5.7 Conclusion

In conclusion, we presented a new framework for 3D reconstruction that significantly improves generalization to new classes with limited training data, and offers multi-view reconstruction for free. Our models take two inputs: the typical image of the object to reconstruct along with a shape prior. Few-shot knowledge consisting of shape models can be used by inputting the average in as the prior. Such a model can then make iterative predictions by using its own output as a prior. Our model requires no novel class images and no retraining. We identified that our model offers far less extremely poor reconstructions than the baseline. We found that this framework peformed well on the multi-view reconstruction task. This finding in particular is surprising given that this model is never trained on multiview. The results here show that explicit categorical information and priors can be a powerful tool in 3D reconstruction.

CHAPTER 6

TIMESERIES ALIGNMENT USING SELF-SUPERVISED VELOCITY FIELDS

6.1 Introduction

Many applications require us to analyze or classify temporally varying data represented as time series. Time series analysis has been used to analyze fashion trends [122], classify a person's activity based on temporal data from a worn sensor [116, 237], or even predict land-use from satellite images of extremely coarse resolutions[128].

A simple but powerful approach for time series classification is to perform nearest neighbors using a distance metric that allows for possible local shifts and scales in the time series. One such distance metric, Dynamic Time Warping (DTW)[15], attains close to state-of-the-art accuracy, and is competitive with all but ensemble-based classifiers[11]. DTW simply computes the optimal alignment of two sequences (allowing for temporal shifts, dilations, and contractions) and returns the absolute difference of the aligned sequences as a distance metric. This classification scheme, which we shall refer to as DTW-kNN, has been shown to be effective on practically every data type in the literature and sees widespread use due to its simplicity and efficacy.

In spite of its efficacy, DTW is computationally expensive: it is *quadratic* in the length of the sequence, making it impractical for classifying long sequences. The dynamic programming approach underlying DTW is also difficult to parallelize effectively. Although optimizations do exist[157, 165, 41, 222, 187], the

fundamental quadratic complexity of full, exact DTW alignment still remains. To be able to analyze large datasets of long sequences over multiple variables, we must therefore look for an alternative that is fundamentally faster, yet just as accurate.

One possible approach is to use *learning* to mimic DTW-based alignment. The hope is that the learnt model, though approximate, might still be just as accurate (if not more) and more importantly, considerably faster. However, many previous attempts at doing so still end up comparing every element of one sequence to every element of the other (e.g., through attention)[71, 127]. Indeed this quadratic pairwise comparison seems central to the alignment task. A recent approach, DTAN, tries to circumvent this by learning to warp each time sequence independently to a canonical reference [214]. However, we find empirically that these alignments are sub-optimal, and lead to *lower* accuracy than the original DTW framework.

In this paper we propose a learning-based alternative to DTW that is *as accurate as* DTW, but is *linear* in the time complexity. We achieve this through two simple but effective ideas. Our first insight is to eschew any quadratic comparison of the elements in the two sequences; instead we simply concatenate the sequences along the channel dimension and use a fully convolutional network. Without quadratic comparisons, it can be difficult for the model to learn good alignments. Our second insight is to make the learning problem substantially simpler through *self-supervision*: we use synthetic warps to generate both data and ground-truth labels for our network to learn. This allows us to train on large datasets *without any human-annotated labels*. Our experiments on a benchmark of time series datasets reveal that this approach even *outperforms* vanilla

DTW while being *an order of magnitude* faster.

Key Contributions: In this work, we present a novel architecture and training scheme which we call *Self-Supervised Alignment via Velocity Fields* (SSAVF). SSAVF trains a neural network that, given two sequences, outputs an aligned version of the first to the second. Importantly, the training is conducted *without either ground-truth class or alignment labels*. We show that SSAVF outperforms vanilla DTW when considered under 1-NN classification, while enjoying the associated parallelisms and optimized hardware of neural networks. This results in both training and inference times which are an order of magnitude faster than standard DTW libraries. We hope that this contribution enables both even widerspread application of timeseries alignment for TSC as well as the integration of sequence alignment into novel neural architectures.

6.2 Related Work

6.2.1 Timeseries Classification

As previously noted, classical machine learning methods and DTW in particular have a rich and relevant history to the problem of TSC. Applications of DTW-based TSC have included early Alzheimer's detection from wearable sensors, voice recognition, and crop classification from Sentinel-2 satellite imagery[196, 137, 52, 128]. A number of methods have proposed optimizations and approximations to DTW, or variants of DTW that admit gradients for backpropagation, such works include[165, 222, 187, 179, 95, 110, 167]. Some

variants soften index selection to admit gradients for backpropagation[41], weight the distance computation to improve performance[84], or use the first-order differences of sequences in addition to the raw values[65, 66]. Other distance-based classifiers include Time Warp Edit distance[125], Move-Split-Merge distance[178], and Complexity Invariant Distance[13].

There are also classifiers based off of dictionaries (e.g. Bag of Patterns[113]), shapelets, or intervals. Finally, ensemble classifiers such as Elastic Ensemble[115] or Collective of Transformation Ensembles[12] offer extremely high performance. We refer the reader to the work of Bagnall et al. for a thorough overview and comparison of the classical TSC algorithm families referred to above[11]. Experiments in this domain are typically validated primarily on the UCR Timeseries Classification Archive[44, 33, 43].

While no deep learning model has shown significant improvements over classical methods, forays into the field should be discussed. Both recurrent and convolutional neural networks have been considered, with convolutional networks generally seeming to offer superior performance[171, 92]. Wang et al. and Jiang both provide surveys comparing deep architectures with classical baselines[213, 85].

6.2.2 Deep Learning on Sequential Data

Timeseries is not the only form of sequential data of interest to the machine learning community. Video (with or without audio), and natural language datasets and applications have garnered intensive interest in recent years[27, 47, 176, 203]. Typically in these applications the sequential objects

(video frames or words) are quite high dimensional, making a concatentation all information impractical. Within such tasks, an image encoder[77] or word representation[131, 130] is often used to generate a sequence of embeddings. This stands in contrast to typical TSC problems, where the dimensionality of the data is much smaller thus obviating the need for sophisticated encoding schemes.

.

6.2.3 Metric Learning

Metric learning methods aim to learn a useful distance function, often a euclidean distance in an embedding space, to operate on data. The triplet, contrastive, and histogram losses[74, 81, 193] are typical examples used in training such systems. Applications include facial recognition, person re-identification, or image retrieval and clustering[232, 211, 193, 174].

6.2.4 Self-Supervised Learning

Self-supervised learning, particularly self-supervised representation learning (SSRL), typically deals with the problem of finding a good feature representation for a dataset using the encoding ability of a neural network. For instance, in computer vision on natural imagery (such as ImageNet[163]), a popular heuristic pretext task is having a network classify the orientation of a rotated image. To do so, the network must learn semantic cues about the physical world, which are useful features for object recognition. Other heuristic meth-

ods include reordering shuffling[142, 111, 135], context/inpainting[47, 151], autoencoding[199] and generative models[20]. Finally, a wave contrastive learning methods has recently dominated the self-supervised learning literature[194, 208, 76, 29, 31, 25, 72]. Initial forays into contrastive learning directly on timeseries include[136, 46].

6.2.5 Temporal/Spatial Transformer Networks

Our method, SSAVF uses a temporal warping network based partially off of the work of Weber et al.[214]. In that work, a *Diffeomorphic Temporal Alignment Network* (DTAN) is proposed. The specifics are discussed in Sec. 6.3, but here it suffices to note that the key part of the DTAN architecture utilizes a Continuous Piecewise Affine-Based (CPAB) velocity field in the time dimension. CPAB fields were proposed and developed in [57, 58]. Other works on temporal warping include: using sequence autoencoders to guide the choice of distance metric[3], learning application-specific temporal invariances with a similar architecture[144] or learning similar invariances in supervised classification. [88] is an example of warping in the *spatial* domain where a network learns how to warp images of birds to match pose.

6.3 Background

Let $x = (x_i)_{i=1}^m$ and $y = (y_i)_{i=1}^m$ denote two univariate timeseries of length m ($x_i, y_i \in \mathbb{R}$). We denote the distance between x_i and y_j, $|x_i - y_j|$, as $c(i, j)$ and the $m \times m$ cost matrix as $C(x, y)$.

6.3.1 DTW

Dynamic time warping computes a matching between x and y. This matching is represented as a set of index pairs $P = \{(i_k, j_k)\}_{k=1}^{M}$, where $m \leq M \leq 2m - 1$ in the general case. DTW seeks a matching that minimizes the total cost $\Sigma_{(i,j)\in P} c(i, j)$, while satisfying the following constraints:

1. All elements of both sequences must have a match, i.e., $\forall \ell \in \{1..m\} \exists i, j \text{s.t} (i, \ell) \in P$ and $(\ell, j) \in P$.
2. The first and last elements are (non-exclusively) matched, i.e., $(1, 1), (m, m) \in P$
3. The mapping between indices is monotonically increasing, i.e., if $i_1 < i_2$ and $(i_1, j_1) \in P$ and $(i_2, j_2) \in P$, then $j_1 \leq j_2$.

The final requirement enforces realism in the warping path, limiting transformations to temporal shifts, contractions, and dilations, but excluding doubling back or folding. This path P can be thought of as a path connecting the upper-left and lower-right ($(1, 1)$ and (m, m)) corners of the cost matrix $C(x, y)$ by taking steps of Chebyshev (chessboard) length 1. The standard approach to finding an optimal path satisfying these constraints is to recursively compute the optimal path for submatrices of $C(x, y)$ to minimize $\Sigma_{(i,j)\in P} c(i, j)$.

The above represents the unconstrained version of DTW. An additional constraint can be added, $|i - j| \leq w \forall (i, j) \in P$. This hyperparameter, w, is known as the *warping window*. The proper selection of w can improve performance significantly[44, 33, 43]. To do this, k-fold cross-validation is performed on the training set, a typical benchmarking standard being to test all w in $\{\lfloor \frac{m}{100} i \rfloor\}_{i=0}^{100}$.

This supervised version of DTW, when combined with 1-NN, is an extremely powerful baseline. For the rest of this work, we will refer to DTW without the window constraint as DTW and DTW with the learned window constraint as DTW-learned and consider them as unsupervised and supervised algorithms respectively.

6.3.2 DTAN

Diffeomorphic temporal alignment networks[214] were introduced as a solution to the problem of multiple sequence alignment (MSA). In MSA, each timeseries (x) of a class is assumed to be the warped version of a signal that is canonically aligned with other members of the same class. As part of learning this canonical intra-class alignment, DTAN computes per-class barycenters of timeseries in the aligned space. These barycenters are shown to minimize variance better than SoftDTW or DTW Barycenter Averaging (DBA)[41, 152].

In DTAN, an input timeseries is fed into a *localization network*, f_{loc}, which outputs θ, a parametrization of a velocity field, $v : \mathbb{R} \to \mathbb{R}$ tesselated across the unit interval (this parametrization primarily involves computing a matrix exponential of an augmented θ to guarantee an invertible affine map, see [57, 58] for details). The warping transformation T^v described by the velocity field v can then be applied to the input sequence and the result integrated to yield a *warped* version. In the original DTAN work, the loss encourages this warp to be smooth while minimizing intra-class variance. We adapt this framework to take two sequences as input and provide an aligning of the first to the second, thus yielding an alternative for DTW, as detailed in the following section.

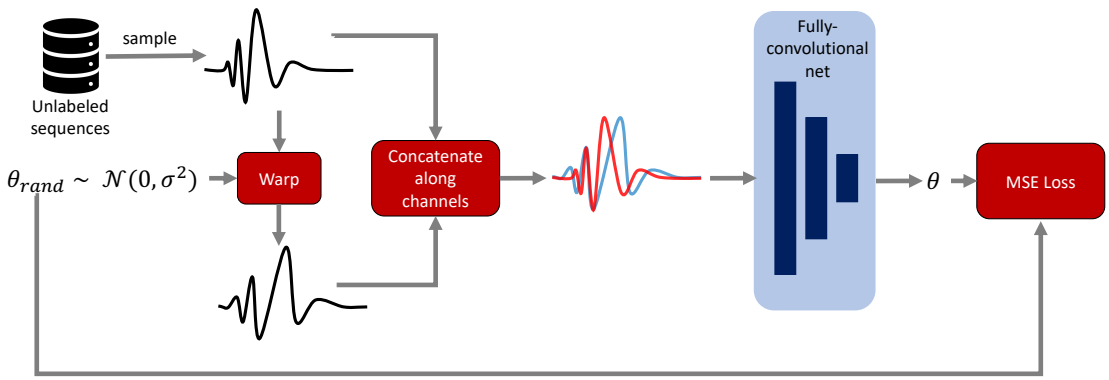

Figure 6.1: The training scheme of SSAVF. Instead of optimizing the aligned distance between two sequences of unknown ground-truth alignment, the loss function is calculated betwen a *known* warping parameterization θ_{rand} and the network output θ. This θ_{rand} is obtained by sampling from a normal distribution with standard deviation σ ($\sigma = 0.1$ in our experiments unless noted otherwise). We find that this training formulation is crucial for learning, and actually results in an alignment accuracy *superior* to DTW for 1-NN classification.

6.4 Method: SSAVF

We consider the problem of adapting DTAN to take in *two* sequences as input and ouput an aligned version of one to the other.

6.4.1 Architecture

Architecturally, our model is identical to DTAN except that instead of the initial temporal convolution being applied on solely the single channel of a solitary input sequence, it takes as input a 2-channel timeseries: the concatenation of the univariate timeseries along the channel dimension. The initial convolution projects the timeseries into an intermediate feature space with high

channel dimensionality, and from there onwards the computation is identical to DTAN, parametrizing the warp to apply to the *first* sequence inputted. Formally, if SSAVF is being trained with batch size b, the input to f_{loc} will be $(x, y) = u \in \mathbb{R}^{b\times 2\times m}$ and the output will be $\theta \in \mathbb{R}^{b\times T}$ where T is the tesselation dimension.

6.4.2 Training and Inference

Most deep alternatives to DTW attempt to approximate the exact DTW alignment/distance through the structure of the output[71] and/or ground truth DTW alignments[127]. We observe, however, that such constraints on the alignment are not neccessarily optimal. The exhaustive pairwise alignment of DTW is of course effective, but a successful neural aligning framework doesn't necessarily need to follow the same constraints (see DTAN as an example of this, as it provides superior barycenters to SoftDTW and DBA).

We found that directly optimizing the aligned euclidean loss of two sequences did not provide a meaningful similarity network that was useful for downstream tasks. We hypothesize that this is due to the highly non-convex nature of the alignment loss. Instead, we drew inspiration from self-supervised representation learning, particularly from methods in computer vision. In such methods, creating synthetic labels (e.g., the orientation of a rotated image) and performing classification has been shown to be a powerful method for learning generalized feature representations[60]. Other such methods, such as jigsaw and colorization, similarly derive a target from a transformation applied to the data itself[142, 243].

In the DTAN architecture, we note that the computation of the final aligned curve from the output θ of f_{loc} is parameter-free, all learning occurs in parametrizing v^θ. This means that standalone losses are not restricted to being computed over the final warped timeseries, $T^v(x)$, but can also be computed on the warp parametrization θ itself. In the case where the warping relationship between the two input sequences is known, a simple mean-squared-error (MSE) loss on θ against the ground-truth might suffice. Since the warping of a sequence given a θ is parameter-free, random warps can be constructed trivially by sampling a θ_{rand} from a random distribution. Given an input sequence, x, we draw $\theta_{rand} \sim N(0, \sigma)$, $(\sigma = 0.1)$ and produce a warped version of x, $x_w = T^{v^{\theta_{rand}}}(x)$. x and x_w are then considered as independent sequences by f_{loc}. The output of $f_{loc}(x, x_w) = \theta$ is compared to the ground-truth θ_{rand} for correctness, using MSE loss. Since our method learns to align two sequences using a velocity field in a self-supervised manner, we denote it *Self-Supervised Alignment via Velocity Fields*, or SSAVF as shorthand. Note that we do not employ an auxiliary smoothing loss as DTAN found necessary.

In subsequent sections we demonstrate that SSAVF learns an effective distance measure, for classification or otherwise *despite having no ground-truth labels or alignments during training*. In fact, remarkably, gradients are never computed between two separate sequences. During training the inputs are always pairs of unaugmented and augmented sequences. The network never calculates the distance between two distinct data points until inference time.

Inference. At inference time, the SSAVF network is treated as a DTW distance function: given two sequences, it returns a distance measure over aligned versions of them. In the case of SSAVF, this distance measure is simply the eu-

clidean distance. Furthermore, the alignment is not the optimized alignment that DTW computes using a dynamic programming algorithm, but instead is a warped version of one input sequence compared to the other.

6.4.3 Training Details

We follow the training protocols of DTAN. 500 batches are run with a batch size of 64 and learning rate of $1e-4$ using an Adam optimizer. A tesselation size of 16 is used to describe the velocity field, v. No data augmentations are applied to the input sequences, apart from the architectural warping that SSAVF does as part of training. The input sequence being warped in SSAVF training is always an unaugmented sample from the original training dataset. We perform experiments on the UCR Time Series Classification Archive[44], considering all datasets of fixed input length with no missing values (113/128).

6.4.4 Time Complexity

The original DTW algorithm runs in $O(MN)$ time [19], where M and N are the lengths of the two sequences being matched (equivalently, $O(N^2)$ if $N \geq M$). There have been recent attempts to improve the running time of the optimal algorithm to $O(N^2/\log\log N)$ [63], but the practical benefits of such a bound are only noticeable for very large N. An approach to parallelize DTW with divide and conquer achieves linear memory, but still maintains quadratic compute time [188].

Other approaches that achieve $O(N)$ time complexity are approximations, ei-

ther learned or otherwise, but suffer from drops in accuracy when compared to original DTW [157, 165, 41, 222, 187]. Our approach attempts to learn the time-sequence warp through a sequence of convolutions and max-pool operations which embed the sequences in a feature space. The continuous-piecewise affine (CPA) based transformation is an integral solved in $O(N)$ time [57, 58], which also makes DTAN an $O(N)$ approach.

Given a 2D convolution kernel size k, an input number of features (depth) of d_i, an output number of filters (depth) of d_o and an input length of n, the convolution operation is $O(kd_id_on)$. The max-pool operation is $O(kd_in)$, where k is the kernel size and d_i is the number of input channels. A fully connected layer operating on an input of length n is $O(d_on)$, where d_o is the dimension of the output sequence. Therefore, a sequence of convolutions and max pool operations have a complexity of $O(LDKN)$, where N is the input length, K is the size of the largest convolutional/pooling kernel, D is the maximum of adjacent $d_i \times d_o$, where d_i and d_o are the input and output number of channels, L is the depth of the network (or number of layers). Given that the network architecture is fixed across datasets, we can treat L, D, K as fixed constants. Therefore, the complexity of the full network is $O(N)$, which is linear in the input sequence.

6.5 Results

We show results on the UCR Time Series Archive[43] (Table 6.1). In Table 6.1 we calculate the average per-dataset classification accuracy of distance metrics in combination with 1-nearest-neighbor. We also show the theoretical time complexity, as well as the actual time taken for inference. Unlike DTW which is

Table 6.1: Nearest-neighbor mean and median accuracies for the UCR Time Series Archive (variable-length and missing-value series removed). The right-most two methods require class labels to calculate similarities while the left-most four do not. The highest value in each category are *bolded*. We further compare the peformances of these methods in Figure 6.2. We reiterate that SSAVF's performance is achieved via self-supervision *with no ground-truth class labels or alignments*.

	Euclidean	DTAN (Unsup.)	DTW	SSAVF (Ours)	DTAN (Sup.)	DTW-learned
1-NN Mean Acc.	69.4%	71.1%	73.0%	**73.7%**	72.2%	**75.1%**
1-NN Median Acc.	72.5%	75.0%	75.5%	**77.5%**	75.1%	**77.9%**
Labels for tuning alignment?	No	No	No	No	Yes	Yes
Complexity w.r.t. Input Length	Linear	Linear	Quadratic	Linear	Linear	Quadratic

quadratic, SSAVF is *linear* in sequence length, leading to a much lower average runtime. DTAN is also linear, but has a substantially lower accuracy.

Note that the two methods on the right (DTAN (Sup.) and DTW-learned) use labeled data to train/tune the alignment module. In contrast, SSAVF is trained without any labels. Tuning the window size in DTW-learned is extremely computationally expensive, thus exacerbating the computational cost. SSAVF achieves a similar median accuracy at a much lower computational cost through a much more lightweight training procedure.

Averaged accuracy statistics give only a partial picture of an algorithm's performance. With this in mind, we construct scatter plots of per-dataset performances between SSAVF and DTAN and DTW, inspired by [85]. Results are shown in Figure 6.2. We see that SSAVF has a winning record against DTW as well as *supervised* DTAN. The one outlier appears in the center of each plot: This marker represents the appliance electrical consumption dataset ACSF1[62], a dataset with extremely high signal frequency. SSAVF constructs velocity fields on a tesselated grid of fixed resolution, meaning the architecture has limited

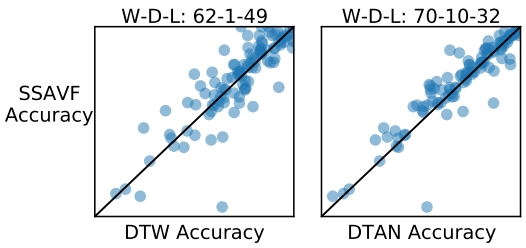

Figure 6.2: A per-dataset comparison of SSAVF vs. DTAN and DTW. Each point in a scatterplot is a dataset of the UCR Time Series Archive. Both axes represent accuracy, the x-axis the accuracy of the method listed at the bottom and the y-axis the accuracy of SSAVF. The gray dashed line is $y = x$, points below this line indicate superior performance by the column method, above by SSAVF. At the top of each scatter is the record between the two datasets, in the form of Wins-Draws-Losses, from the perspective of SSAVF.

ability to align high frequency signals. For the majority of datasets, however, these plots indicate that SSAVF offers a robust choice of unsupervised *linear-time* distance measure which performs consistently well when compared to existing methods.

6.6 Analysis

6.6.1 Induced Distance

We compare the "distance" induced by SSAVF to Euclidean and DTW distances. Recall that SSAVF demonstrates classification ability far superior to Euclidean 1-NN and slightly better than vanilla DTW. In Figure 6.3, we observe that the similarity induced by SSAVF in fact is much more correlated with the Euclidean distance than the DTW distance. This stands in contrast to what one might ex-

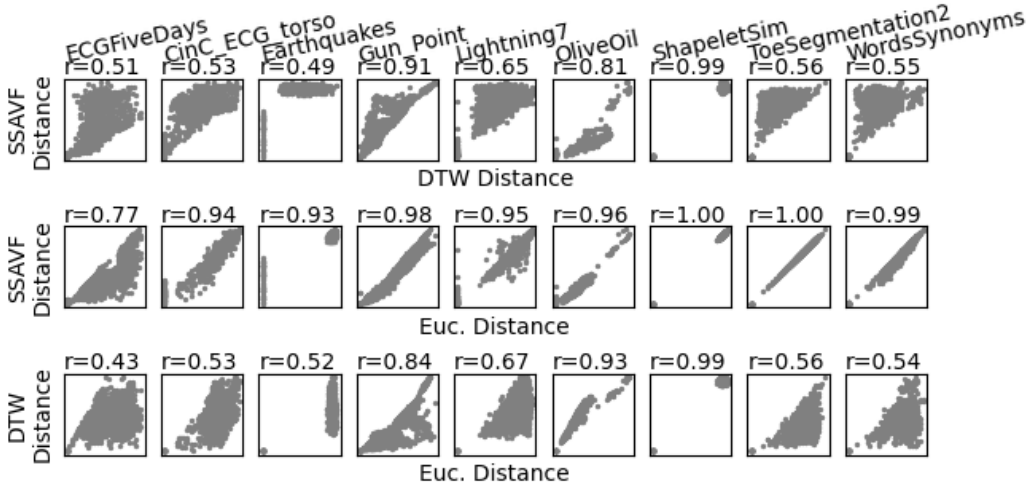

Figure 6.3: Comparisons and correlations of Euclidean distance, DTW, and SSAVF-induced "distance". Datasets selected at random. The set of plots in each column is for a given dataset, each row shows a distance-distance pair. We observe that SSAVF is much more strongly correlated with the *Euclidean* distance than with DTW, despite classification performance more similar to the latter.

pect, that only by precisely capturing the DTW distance computation would a model be able to match its performance and that the performance of DTW would serve as an upper bound to a model trained in the manner of SSAVF. On the other hand, this does give insight into how SSAVF can indeed *surpass* the performance of vanilla DTW: if it had learned the exact metric then performance would be identical. We visualize a sample alignment from our model in Figure 6.4.

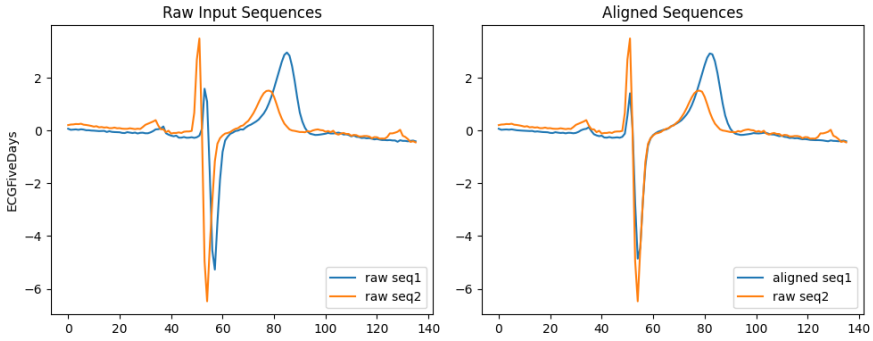

Figure 6.4: Sample alignments from our model. We see that despite being trained with a loss over suggested and synthetic warps instead of raw values, good alignment is achieved between sequences. For example, on the ECGFiveDays dataset, the dip of the heartbeat in the first sequence becomes perfectly aligned to that of the second.

6.6.2 t-SNE

We observe that SSAVF is particularly effective on a specific type of data from the UCR Archive: electrocardiogram (ECG) readings. Here SSAVF has mean accuracy of 87.7%, a percentage point higher than any other method. We observe the clustering abilities of SSAVF in Figure 6.5. SSAVF forms extremely coherent clusters: for the majority of datasets, classes are partitioned almost perfectly. Visually the cluster quality seems to exceed that of all other methods, even DTAN which benefits from label supervision during training.

6.7 Conclusion

In summary, we have presented a new deep metric learning approach for time-series and demonstrated its classification ability. At its core, our approach is a learned neural network that aligns time series with complexity linear in the

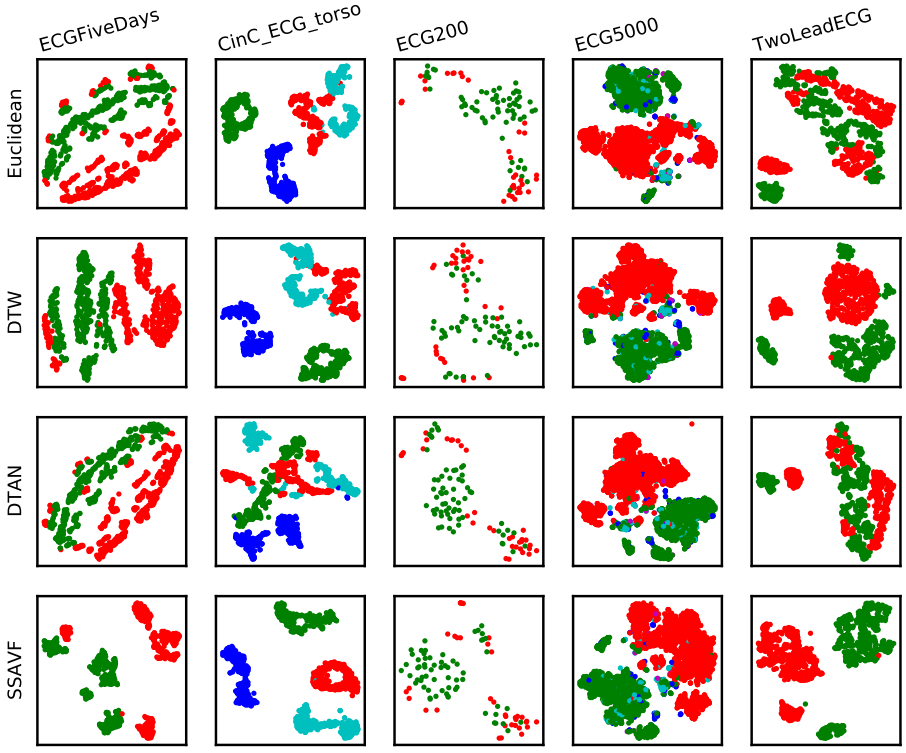

Figure 6.5: t-SNE[118] plots for considered methods on all traditional ECG datasets from the UCR Archive. We see that SSAVF is particularly effective for this datatype, forming very coherent clusters.

length of the sequence. This neural network is trained without supervision, using randomly warped time series as a self-supervisory signal. The resulting model is as accurate as DTW while being substantially faster, and is also more accurate than other fast, learned approximations. In the future, we aim to expand this model to be generalizable across timeseries, providing all-purpose utility as ImageNet-trained models have historically given to computer vision.

CHAPTER 7

CONCLUSION

I conclude with a discussion of several ways in which I anticipate this field evolving, and how my presented work will play a role.

7.1 Avenues of General Improvement

The results of Chapter 3 are extremely exciting, as they reveal the possibility of intelligently improving pretrained features without any additional knowledge outside of the new images themselves. This practically free lunch demonstrates that network embeddings contain more information than the head network has learned how to exploit.

The interaction of various networks when ensembled in such a manner is ripe for exploration. Different networks tend towards different biases and errors, and finding the optimal set of networks to use on a given problem would be of great interest (such selection would play into the work of Chapter 4). Furthermore, the necessary strength of the ensembled networks is similarly unexplored. In our work we used exclusively networks already known to provide good representations. What if the representations are similarly trained but of lower quality? Or even random? This self-supervised ensembling opens up a whole new possible sub-field of intrigue that could dramatically change how we use pretrained networks in application.

From a technical perspective, this method also opens the door to more tailored methods of feature improvement, where self-supervised head networks can be trained on top of a fixed set of frozen feature extractors. An example of

the above might be shaping the representation space's distribution using a prior or imposing another such structure. Additionally, we did not employ image augmentations in our work. Learning a mapping of the original representation space to a feature space invariant to one or multiple augmentations is a promosing avenue. Such augmentations might be able to be identified automatically to preserve the universal applicability of our method.

7.2 Expert Selection

The field of expert selection is nascent, with the initial foray[4] only being published two years ago and with little academic interest thus far. Despite this current lack of general interest, there are reasons to believe that this field offers long-term promise. As the use of computer vision continues to proliferate, so does the number of computer vision datasets. As time goes on, it is increasingly rare to work on a collection of imagery dramatically different from any before. While labeling is expensive, the accumulation of such labeling in the community over time will help begin to desparsify the dataset space and make expert selection pretraining increasingly valuable in practice.

There are also multiple ways in which the notion of expert selection can be adapted for higher impact. First off, only image data was considered in this work; the possibility of application to other data types is fully unexplored. Next, the problem of Chapter 4 can be construed in the more general case as *choosing how to perform pretraining for a given task*. Here we have focused solely on supervised pretraining on the entirety of a single other dataset, but more choices exist. For example, *per-example* selection could be made. In this setting an algorithm

would flexibly select a subset of an image collection to use in pretraining instead of a single group of images. Furthermore, leveraging the insights of Chapter 2, methods of self-supervised training could be automatically selected by similar algorithms. All of these directions would be valuable automatic machine learning contributions which would help improve the practice of deep learning by non-specialists.

7.3 CLIP and Other Web Supervision

While the story of computer vision up until now has been unwritable without mention of ImageNet pretraining; a new method, called Contrastive Language-Image Pretraining (CLIP)[155], offers an extremely exciting alternative that has begun seeing widespread use in the past few months. CLIP is a method of learning a joint language-vision embedding space, trained in the cited paper on the captions of 400 million online images. Classification across *any* set of classes on a novel image can then be performed by measuring the similarity of the corresponding visual embedding with language embeddings of the class name textual embeddings. This process leverages the vast diversity of the data seen by a CLIP network in pretraining. CLIP has achieved state-of-the-art zero-shot classification on a wide variety of tasks and has seen many applications such as video understanding or image and shape generation/manipulation[6, 166, 227]. Such models offer a very intriguing option of *zero* retraining being needed, not even a head network. This is one of the most fundamental and exciting advances in the last several years of computer vision. With computational resources increasing, web supervision is becoming increasingly practical for at least a subset of research groups. While there is great promise in the power and

application of such methods, there are concerns that will need to be addressed. It is well documented that biases (both societal and statistical) can exist in both curated and uncurated datasets[22, 207], and this problem is very prominent in web content. Web-trained models must be applied carefully to not suffer from these biases, and the study of debiasing the feature space in transfer learning (particularly debiasing in a dataset-specific manner) has been largely unstudied.

7.4 Recreating Success on Other Data Modes

This thesis has focused predominantly on computer vision, discussing the remarkable transferability of a single family of pretrained vision models, the inception of which jumpstarted the field of deep learning and computer vision as a whole. Natural language processing has recently enjoyed a similar phenomenon, with models such as BERT and GPT[47, 156] providing widespread out-of-the-box applicability and finetuning. These successes have both been immediately followed by huge waves of application where the vast majority of benchmark tasks behave like proverbial nails to the hammer of the pretrained model. For many data types, this event has yet to occur. I discussed in Chapter 6 the promising beginnings of self-supervision on timeseries data. In time I hope to create a generalizable model that can be as broadly applied to timeseries data as the pretrained models mentioned throughout this work. Audio is another domain with increasing interest where a generic pretrained feature extractor could yield many applications in rapid succession.

Even established fields such as reinforcement learning and 3D ”vision“ do

not yet have any network that comes as close to a panacea as large vision and language pretrained models. These domains represent a small fraction of the possibilities for deep learning. The physical sciences offer a vast variety of data types. Just from a classification perspective: numerical specifications, schematics (e.g. blueprints or CAD files), chemical concentration measurements, networks, haptics, olfactory data, and large-scale systems (e.g. finite element method scenarios) exist and as of now have received relatively little attention from the perspective of generalizable models.

Finally, classification is perhaps the simplest machine learning task; however large the input data the ultimate goal is to reduce it down to a precise vector with as many dimensions as the number of classes (or another similar form). Other vision problems, such as segmentation, detection, depth estimation, etc., can be, and typically are, formulated as granular classification problems. Overall, a surprising amount of current vision problems reduce down to some number of classification tasks where the goal is simplification. Data *generation* is a task in the opposite direction, with the goal of making a complex structure from a simple goal or prompt. Current examples of practical applications in computer vision include domain translation (e.g. sketch to image, image blending) and attribute manipulation/photograph editing (e.g. de-aging, rain removal, super resolution). As of now, there is very little model unification across the sub-fields as there has been in predictive (classification) tasks; text-to-image (possibly leveraging CLIP as in [6]) is perhaps the most related current work. Language, on the other hand, already has a near-universal text generator in the previously mentioned GPT model. These language generators have even been adapted to tasks such as fine-grained protein generation[119]. Reliable, customizable, universal generators being developed for vision would

be extremely impactful itself. If such generation could be achieved on domains such as schematics or networks, it could fundamentally change how many engineering tasks are performed.

BIBLIOGRAPHY

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Salisu Mamman Abdulrahman, Pavel Brazdil, Jan N van Rijn, and Joaquin Vanschoren. Speeding up algorithm selection using average ranking and active testing by introducing runtime. *Machine learning*, 107(1):79–108, 2018.

[3] Abubakar Abid and James Y Zou. Learning a warping distance from unlabeled time series using sequence autoencoders. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[4] Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C. Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[5] Eliot Andres. Kaggle past solutions.

[6] Anonymous. DiffusionCLIP: Text-guided image manipulation using diffusion models. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. under review.

[7] Guilherme Aresta, Teresa Araújo, Scotty Kwok, Sai Saketh Chennamsetty, Mohammed Safwan, Varghese Alex, Bahram Marami, Marcel Prastawa, Monica Chan, Michael Donovan, et al. Bach: Grand challenge on breast cancer histology images. *Medical image analysis*, 56:122–139, 2019.

[8] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit reg-

ularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32:7413–7424, 2019.

[9] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pages 8141–8150, 2019.

[10] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *European conference on computer vision*, pages 584–599. Springer, 2014.

[11] Anthony Bagnall, Aaron Bostrom, James Large, and Jason Lines. The great time series classification bake off: An experimental evaluation of recently proposed algorithms. extended version, 2016.

[12] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[13] Gustavo EAPA Batista, Eamonn J Keogh, Oben Moses Tataw, and Vinicius MA De Souza. Cid: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, 28(3):634–669, 2014.

[14] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[15] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

[16] Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould. Dynamic image networks for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3034–3042, 2016.

[17] Piotr Bojanowski and Armand Joulin. Unsupervised learning by predicting noise. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 517–526. JMLR. org, 2017.

[18] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – min-

ing discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[19] Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. *CoRR*, abs/1502.01063, 2015.

[20] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2019.

[21] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[22] Kaylee Burns, Lisa Anne Hendricks, Trevor Darrell, and Anna Rohrbach. Women also snowboard: Overcoming bias in captioning models. *CoRR*, abs/1803.09797, 2018.

[23] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

[24] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[25] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments, 2021.

[26] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[27] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. Vggsound: A large-scale audio-visual dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 721–725. IEEE, 2020.

[28] Liang Chen, Paul Bentley, Kensaku Mori, Kazunari Misawa, Michitaka

Fujiwara, and Daniel Rueckert. Self-supervised learning for medical image analysis using image context restoration. *Medical image analysis*, 58:101539, 2019.

[29] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[30] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22243–22255. Curran Associates, Inc., 2020.

[31] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning, 2020.

[32] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

[33] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive, July 2015. www.cs.ucr.edu/ eamonn/time_series_data/.

[34] Ian Cherabier, Johannes L Schonberger, Martin R Oswald, Marc Pollefeys, and Andreas Geiger. Learning priors for semantic 3d reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 314–330, 2018.

[35] François Chollet et al. Keras. https://keras.io, 2015.

[36] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[37] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014.

[38] Noel Codella, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi,

Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368*, 2019.

[39] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[40] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4109–4118, 2018.

[41] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International Conference on Machine Learning*, pages 894–903. PMLR, 2017.

[42] Christoph Dalitz. Reject options and confidence measures for knn classifiers. *Schriftenreihe des Fachbereichs Elektrotechnik und Informatik Hochschule Niederrhein*, 8:16–38, 2009.

[43] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[44] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. The ucr time series classification archive, October 2018. https://www.cs.ucr.edu/ eamonn/-time_series_data_2018/.

[45] Virginia R de Sa. Learning classification with unlabeled data. In *Advances in neural information processing systems*, pages 112–119, 1994.

[46] Shohreh Deldari, Daniel V. Smith, Hao Xue, and Flora D. Salim. Time-series change point detection with self-supervised contrastive predictive coding. *CoRR*, abs/2011.14097, 2020.

[47] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[48] Luc Devroye, Laszlo Gyorfi, and Gabor Lugosi. Consistency of the k-nearest neighbor rule. In *A Probabilistic Theory of Pattern Recognition*, pages 169–185. Springer, 1996.

[49] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[50] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[51] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[52] Matthias Drusch, Umberto Del Bello, Sébastien Carlier, Olivier Colin, Veronica Fernandez, Ferran Gascon, Bianca Hoersch, Claudia Isola, Paolo Laberinti, Philippe Martimort, et al. Sentinel-2: Esa's optical high-resolution mission for gmes operational services. *Remote sensing of Environment*, 120:25–36, 2012.

[53] Jiali Duan, Yen-Liang Lin, Son Tran, Larry S Davis, and C-C Jay Kuo. Slade: A self-training framework for distance metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9644–9653, 2021.

[54] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[55] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning, 2019.

[56] Patrick Esser, Ekaterina Sutter, and Björn Ommer. A variational u-net for conditional appearance and shape generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8857–8866, 2018.

[57] Oren Freifeld, Soren Hauberg, Kayhan Batmanghelich, and John W.

Fisher, III. Highly-expressive spaces of well-behaved transformations: Keeping it simple. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[58] Oren Freifeld, Søren Hauberg, Kayhan Batmanghelich, and Jonn W. Fisher. Transformations based on continuous piecewise-affine velocity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2496–2509, 2017.

[59] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 32(8):1362–1376, 2010.

[60] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

[61] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

[62] Christophe Gisler, Antonio Ridi, Damien Zufferey, Omar Abou Khaled, and Jean Hennebert. Appliance consumption signature database and recognition test protocols. In *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)*, pages 336–341. IEEE, 2013.

[63] Omer Gold and Micha Sharir. Dynamic time warping and geometric edit distance: Breaking the quadratic barrier. *ACM Trans. Algorithms*, 14(4), August 2018.

[64] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[65] Tomasz Górecki and Maciej Łuczak. Using derivatives in time series classification. *Data Mining and Knowledge Discovery*, 26(2):310–331, 2013.

[66] Tomasz Górecki and Maciej Łuczak. Non-isometric transforms in time series classification using dtw. *Knowledge-based systems*, 61:98–108, 2014.

[67] Ross Goroshin, Michael F Mathieu, and Yann LeCun. Learning to linearize under uncertainty. In *Advances in Neural Information Processing Systems*, pages 1234–1242, 2015.

[68] Priya Goyal, Quentin Duval, Jeremy Reizenstein, Matthew Leavitt, Min Xu, Benjamin Lefaudeux, Mannat Singh, Vinicius Reis, Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Ishan Misra. Vissl. https://github.com/facebookresearch/vissl, 2021.

[69] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[70] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. *arXiv preprint arXiv:1905.01235*, 2019.

[71] Josif Grabocka and Lars Schmidt-Thieme. Neuralwarp: Time-series similarity with warping networks. *arXiv preprint arXiv:1812.08306*, 2018.

[72] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[73] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *CVPR*, 2018.

[74] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[75] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[76] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[77] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[78] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[79] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[80] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[81] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015.

[82] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

[83] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[84] Young-Seon Jeong, Myong K Jeong, and Olufemi A Omitaomu. Weighted dynamic time warping for time series classification. *Pattern recognition*, 44(9):2231–2240, 2011.

[85] Weiwei Jiang. Time series classification: nearest neighbor versus deep learning models. *SN Applied Sciences*, 2(4):1–17, 2020.

[86] Hadi S Jomaa, Lars Schmidt-Thieme, and Josif Grabocka. Dataset2vec: Learning dataset meta-features. *arXiv preprint arXiv:1905.11063*, 2019.

[87] Michiel Kallenberg, Kersten Petersen, Mads Nielsen, Andrew Y Ng, Pengfei Diao, Christian Igel, Celine M Vachon, Katharina Holland, Rikke Rass Winkel, Nico Karssemeijer, et al. Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring. *IEEE transactions on medical imaging*, 35(5):1322–1331, 2016.

[88] Angjoo Kanazawa, David W. Jacobs, and Manmohan Chandraker. Warpnet: Weakly supervised matching for single-view reconstruction, 2016.

[89] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

[90] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Advances in neural information processing systems*, pages 365–376, 2017.

[91] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. 2017.

[92] Fazle Karim, Somshubra Majumdar, and Houshang Darabi. Insights into lstm fully convolutional networks for time series classification. *IEEE Access*, 7:67718–67725, 2019.

[93] Jakob Nikolas Kather, Cleo-Aron Weis, Francesco Bianconi, Susanne M Melchers, Lothar R Schad, Timo Gaiser, Alexander Marx, and Frank Gerrit Zöllner. Multi-class texture analysis in colorectal cancer histology. *Scientific reports*, 6:27988, 2016.

[94] Hiroharu Kato and Tatsuya Harada. Learning view priors for single-view 3d reconstruction. *CoRR*, abs/1811.10719, 2018.

[95] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005.

[96] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2021.

[97] Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. 2004.

[98] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[99] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Large scale learning of gen-

eral visual representations for transfer. *arXiv preprint arXiv:1912.11370*, 2019.

[100] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[101] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[102] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization.

[103] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[104] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[105] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000.

[106] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[107] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision*, pages 577–593. Springer, 2016.

[108] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks.

[109] Rui Leite, Pavel Brazdil, and Joaquin Vanschoren. Selecting classification algorithms with active testing. In *International workshop on machine learning and data mining in pattern recognition*, pages 117–131. Springer, 2012.

[110] Daniel Lemire. Faster retrieval with a two-pass dynamic-time-warping lower bound. *Pattern recognition*, 42(9):2169–2180, 2009.

[111] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[112] Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S. Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. Enhanced convolutional neural tangent kernels, 2019.

[113] Jessica Lin, Rohan Khade, and Yuan Li. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, 39(2):287–315, 2012.

[114] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[115] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

[116] Li Liu, Yuxin Peng, Ming Liu, and Zigang Huang. Sensor-based human activity recognition system with a multilayered model using time series shapelets. *Knowledge-Based Systems*, 90:138–152, 2015.

[117] Alex X Lu, Oren Z Kraus, Sam Cooper, and Alan M Moses. Learning unsupervised feature representations for single cell microscopy images with paired cell inpainting. *PLoS computational biology*, 15(9):e1007348, 2019.

[118] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[119] Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

[120] 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. *CoRR*, abs/1805.00932, 2018.

[121] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[122] Utkarsh Mall, Kevin Matzen, Bharath Hariharan, Noah Snavely, and Kavita Bala. GeoStyle: Discovering fashion trends and events. In *ICCV*, 2019.

[123] The man Ezra Cornell himself. First person to notify me of this citation gets a $10 cornell dairy gift card. https://scl.cornell.edu/residential-life/dining/eateries-menus/cafes-food-courts-coffeehouses/cornell-dairy-bar, 1865.

[124] Stephan Mandt, Jasper Snoek, Tim Salimans, Sebastian Nowozin, and Rodolphe Jenatton. Hydra: Preserving ensemble diversity for model distillation.

[125] Pierre-François Marteau. Time warp edit distance with stiffness adjustment for time series matching. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):306–318, 2008.

[126] Pyry Matikainen, Rahul Sukthankar, and Martial Hebert. Model recommendation for action recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2256–2263. IEEE, 2012.

[127] Shinnosuke Matsuo, Xiaomeng Wu, Gantugs Atarsaikhan, Akisato Kimura, Kunio Kashino, Brian Kenji Iwana, and Seiichi Uchida. Attention to warp: Deep metric learning for multivariate time series. *arXiv preprint arXiv:2103.15074*, 2021.

[128] Victor Maus, Gilberto Câmara, Ricardo Cartaxo, Alber Sanchez, Fernando M. Ramos, and Gilberto R. de Queiroz. A time-weighted dynamic time warping method for land-use and land-cover mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(8):3729–3739, 2016.

[129] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

[130] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[131] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.

[132] Erik G Miller, Nicholas E Matsakis, and Paul A Viola. Learning from one example through shared densities on transforms. In *CVPR*, 2000.

[133] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. *arXiv preprint arXiv:1912.01991*, 2019.

[134] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016.

[135] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016.

[136] Mostafa Neo Mohsenvand, Mohammad Rasool Izadi, and Pattie Maes. Contrastive representation learning for electroencephalogram classification. In Emily Alsentzer, Matthew B. A. McDermott, Fabian Falck, Suproteem K. Sarkar, Subhrajit Roy, and Stephanie L. Hyland, editors, *Proceedings of the Machine Learning for Health NeurIPS Workshop*, volume 136 of *Proceedings of Machine Learning Research*, pages 238–253. PMLR, 11 Dec 2020.

[137] Lindasalwa Muda, Mumtaj Begam, and I. Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *CoRR*, abs/1003.4083, 2010.

[138] Stefan Munder and Dariu M Gavrila. An experimental study on pedestrian classification. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1863–1868, 2006.

[139] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[140] Cuong V. Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. Leep: A new measure to evaluate transferability of learned representations, 2020.

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

[142] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representions by solving jigsaw puzzles. In *ECCV*, 2016.

[143] Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. Neural tangents: Fast and easy infinite neural networks in python. In *International Conference on Learning Representations*, 2020.

[144] Jeeheh Oh, Jiaxuan Wang, and Jenna Wiens. Learning to exploit invariances in clinical time-series data using sequence transformer networks. In Finale Doshi-Velez, Jim Fackler, Ken Jung, David Kale, Rajesh Ranganath, Byron Wallace, and Jenna Wiens, editors, *Proceedings of the 3rd Machine Learning for Healthcare Conference*, volume 85 of *Proceedings of Machine Learning Research*, pages 332–347, Palo Alto, California, 17–18 Aug 2018. PMLR.

[145] Wei Ouyang, Casper Winsnes, Martin Hjelmare, Anthony Cesnik, Lovisa kesson, Hao Xu, Devin Sullivan, Shubin Dai, Jun Lan, Park Jinmo, Shaikat Mahmood Galib, Christof Henkel, Kevin Hwang, Dmytro Poplavskiy, Bojan Tunguz, Russel Wolfinger, Yinzheng Gu, Chuanpeng Li, Jinbin Xie, and Emma Lundberg. Analysis of the human protein atlas image classification competition. *Nature Methods*, 16:1254–1261, 12 2019.

[146] Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. Ambient sound provides supervision for visual learning. In *European conference on computer vision*, pages 801–816. Springer, 2016.

[147] Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

[148] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[149] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2701–2710, 2017.

[150] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[151] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[152] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3):678–693, 2011.

[153] Cheng Perng Phoo and Bharath Hariharan. Self-training for few-shot transfer across extreme task differences. *arXiv preprint arXiv:2010.07734*, 2020.

[154] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420. IEEE, 2009.

[155] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[156] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[157] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 262–270, 2012.

[158] S-A Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, 2017.

[159] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[160] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016.

[161] Stephane Ross, Daniel Munoz, Martial Hebert, and J. Andrew Bagnell. Learning message-passing inference machines for structured prediction. In *CVPR*, 2011.

[162] Yong Rui, Thomas S Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of visual communication and image representation*, 10(1):39–62, 1999.

[163] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[164] S. Saha and S. Bandyopadhyay. Unsupervised pixel classification in satellite imagery using a new multiobjective symmetry based clustering approach. In *TENCON 2008 - 2008 IEEE Region 10 Conference*, pages 1–6, 2008.

[165] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

[166] Aditya Sanghi, Hang Chu, Joseph G. Lambourne, Ye Wang, Chin-Yi Cheng, and Marco Fumero. Clip-forge: Towards zero-shot text-to-shape generation, 2021.

[167] Yilin Shen, Yanping Chen, Eamonn Keogh, and Hongxia Jin. Accelerating time series searching with large uniform scaling. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 234–242. SIAM, 2018.

[168] Assaf Shocher, Nadav Cohen, and Michal Irani. "zero-shot" super-resolution using deep internal learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3118–3126, 2018.

[169] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[170] Vincent Sitzmann, Eric R Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *arXiv preprint arXiv:2006.09662*, 2020.

[171] Denis Smirnov and Engelbert Mephu Nguifo. Time series classification with recurrent neural networks. *Advanced Analytics and Learning on Temporal Data*, 8, 2018.

[172] Michael R Smith, Logan Mitchell, Christophe Giraud-Carrier, and Tony Martinez. Recommending learning algorithms and their associated hyperparameters. *arXiv preprint arXiv:1407.1890*, 2014.

[173] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017.

[174] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 1857–1865, 2016.

[175] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[176] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[177] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

[178] Alexandra Stefan, Vassilis Athitsos, and Gautam Das. The move-split-merge metric for time series. *IEEE transactions on Knowledge and Data Engineering*, 25(6):1425–1438, 2012.

[179] Peter Steffen, Robert Giegerich, and Mathieu Giraud. Gpu parallelization of algebraic dynamic programming. In *International Conference on Parallel Processing and Applied Mathematics*, pages 290–299. Springer, 2009.

[180] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? *arXiv preprint arXiv:1910.03560*, 2019.

[181] Gencer Sumbul, Marcela Charfuelan, Begüm Demir, and Volker Markl. Bigearthnet: A large-scale benchmark archive for remote sensing image understanding.

[182] Yu Sun, Xiaolong Wang, Liu Zhuang, John Miller, Moritz Hardt, and Alexei A. Efros. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020.

[183] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017.

[184] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019.

[185] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

[186] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[187] Christopher Tralie and Elizabeth Dempsey. Exact, parallelizable dynamic time warping alignment with linear memory. *arXiv preprint arXiv:2008.02734*, 2020.

[188] Christopher J. Tralie and Elizabeth Dempsey. Exact, parallelizable dynamic time warping alignment with linear memory. *CoRR*, abs/2008.02734, 2020.

[189] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5:180161, 2018.

[190] Zhuowen Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, pages 1–8, 2008.

[191] Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *CVPR*, 2018.

[192] Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7064–7073, 2017.

[193] Evgeniya Ustinova and Victor Lempitsky. Learning deep embeddings with histogram loss. *arXiv preprint arXiv:1611.00822*, 2016.

[194] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019.

[195] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[196] Ramachandran Varatharajan, Gunasekaran Manogaran, Malarvizhi Kumar Priyan, and Revathi Sundarasekar. Wearable sensor devices for early detection of alzheimer disease using dynamic time warping algorithm. *Cluster Computing*, 21(1):681–690, 2018.

[197] Alfredo Vellido. The importance of interpretability and visualization in machine learning for applications in medicine and health care. *Neural Computing and Applications*, pages 1–15, 2019.

[198] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

[199] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

[200] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[201] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851. Springer, 2016.

[202] Bram Wallace and Bharath Hariharan. Extending and analyzing self-supervised learning across domains, 2020.

[203] Luyu Wang, Pauline Luc, Adria Recasens, Jean-Baptiste Alayrac, and Aaron van den Oord. Multimodal self-supervised learning of general audio representations. *arXiv preprint arXiv:2104.12807*, 2021.

[204] Meng Wang, Lingjing Wang, and Yi Fang. 3densinet: A robust neural network architecture towards 3d volumetric object prediction from 2d image. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 961–969. ACM, 2017.

[205] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018.

[206] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017.

[207] Tianlu Wang, Xi Victoria Lin, Nazneen Fatema Rajani, Bryan McCann, Vicente Ordonez, and Caiming Xiong. Double-hard debias: Tailoring word embeddings for gender bias mitigation, 2020.

[208] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR, 13–18 Jul 2020.

[209] Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 3dn: 3d deformation network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1038–1046, 2019.

[210] Xudong Wang, Zhaowei Cai, Dashan Gao, and Nuno Vasconcelos. Towards universal object detection by domain attention. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[211] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5022–5030, 2019.

[212] Yu-Xiong Wang and Martial Hebert. Model recommendation: Generating object detectors from few samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1619–1628, 2015.

[213] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[214] Ron Shapira Weber, Matan Eyal, Nicki Skafte Detlefsen, Oren Shriki, and Oren Freifeld. Diffeomorphic temporal alignment nets. In *NeurIPS*, pages 6570–6581, 2019.

[215] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010.

[216] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017.

[217] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pages 82–90, 2016.

[218] Jiajun Wu, Chengkai Zhang, Xiuming Zhang, Zhoutong Zhang, William T. Freeman, and Joshua B. Tenenbaum. Learning shape priors for single-view 3d completion and reconstruction. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[219] Yanzhao Wu, Ling Liu, Zhongwei Xie, Ka-Ho Chow, and Wenqi Wei. Boosting ensemble accuracy by revisiting ensemble diversity metrics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16469–16477, June 2021.

[220] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.

[221] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 75–82, 2014.

[222] Limin Xiao, Yao Zheng, Wenqi Tang, Guangchao Yao, and Li Ruan. Parallelizing dynamic time warping algorithm using prefix computations on gpu. In *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing*, pages 294–299. IEEE, 2013.

[223] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification, 2020.

[224] Yu xiong Wang, Ross Girshick, Martial Herbert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, 2018.

[225] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[226] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604. Springer, 2020.

[227] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. Videoclip: Contrastive pre-training for zero-shot video-text understanding, 2021.

[228] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *NeurIPS*, pages 1696–1704, 2016.

[229] Guandao Yang, Yin Cui, Serge Belongie, and Bharath Hariharan. Learning single-view 3d reconstruction with limited pose supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 86–101, 2018.

[230] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*, 2018.

[231] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279. ACM, 2010.

[232] Hantao Yao, Shiliang Zhang, Richang Hong, Yongdong Zhang, Changsheng Xu, and Qi Tian. Deep representation learning with part loss for person re-identification. *IEEE Transactions on Image Processing*, 28(6):2860–2871, 2019.

[233] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction, 2020.

[234] Amir Zadeh, Yao-Chong Lim, Paul Pu Liang, and Louis-Philippe Morency. Variational auto-decoder: A method for neural generative modeling from incomplete data. *arXiv preprint arXiv:1903.00840*, 2019.

[235] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018.

[236] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *CoRR*, abs/2103.03230, 2021.

[237] Tahmina Zebin, Matthew Sperrin, Niels Peek, and Alexander J Casson. Human activity recognition from inertial sensor time-series using batch normalized deep lstm recurrent networks. In *2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, pages 1–4. IEEE, 2018.

[238] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[239] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE international conference on computer vision*, pages 1476–1485, 2019.

[240] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. The visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

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

[242] Peng Zhang, Jiuling Wang, Ali Farhadi, Martial Hebert, and Devi Parikh. Predicting failures of vision systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3566–3573, 2014.

[243] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[244] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017.

[245] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[246] Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Josh Tenenbaum, Bill Freeman, and Jiajun Wu. Learning to reconstruct shapes from unseen classes. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2257–2268. Curran Associates, Inc., 2018.

[247] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[248] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017.