Self-Supervised Pretraining Improves Self-Supervised Pretraining

Colorado J Reed*[1] Xiangyu Yue*[1] Ani Nrusimha[1] Sayna Ebrahimi[1]
Vivek Vijaykumar[3] Richard Mao[1] Bo Li[1] Shanghang Zhang[1] Devin Guillory[1]
Sean Metzger[1,2] Kurt Keutzer[1] Trevor Darrell[1]

[1]UC Berkeley, [2]UCSF, [3]Georgia Tech

*equal contribution, correspondence to cjrd@berkeley.edu

Abstract

While self-supervised pretraining has proven beneficial for many computer vision tasks, it requires expensive and lengthy computation, large amounts of data, and is sensitive to data augmentation. Prior work demonstrates that models pretrained on datasets dissimilar to their target data, such as chest X-ray models trained on ImageNet, underperform models trained from scratch. Users that lack the resources to pretrain must use existing models with lower performance. This paper explores Hierarchical PreTraining (HPT), which decreases convergence time and improves accuracy by initializing the pretraining process with an existing pretrained model. Through experimentation on 16 diverse vision datasets, we show HPT converges up to 80× faster, improves accuracy across tasks, and improves the robustness of the self-supervised pretraining process to changes in the image augmentation policy or amount of pretraining data. Taken together, HPT provides a simple framework for obtaining better pretrained representations with less computational resources.

1. Introduction

Recently, self-supervised pretraining – an unsupervised pretraining method that self-labels data to learn salient feature representations – has outperformed supervised pretraining in an increasing number of computer vision applications [5, 7, 4]. These advances come from *instance contrastive learning*, where a model is trained to identify visually augmented images that originated from the same image from a set [14, 59]. Typically, self-supervised pretraining uses unlabeled *source* data to pretrain a network that will be *transferred* to a supervised training process on a *target* dataset. Self-supervised pretraining is particularly useful when labeling is costly, such as in medical and satellite imaging [53, 8].

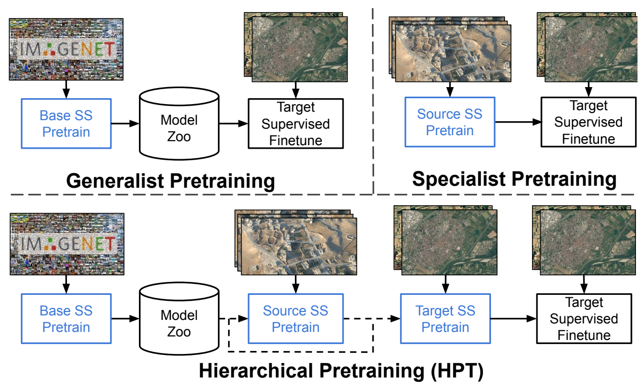

Figure 1. *Methods of using self-supervision.* The top row are the two common prior approaches to using self-supervised (SS) pretraining. In *Generalist Pretraining*, a large, general, *base* dataset is used for pretraining, e.g. ImageNet. In *Specialist Pretraining*, a large, specialized *source* dataset is collected and used for pretraining, e.g. aerial images. In this paper, we explore *Hierarchical Pre-Training* (HPT), which sequentially pretrains on datasets that are similar to the target data, thus providing the improved performance of specialist pretraining while leveraging existing models.

However, self-supervised pretraining requires long training time on large datasets, e.g. SimCLR [5] showed improved performance out to 3200 epochs on ImageNet's 1.2 million images [51]. In addition, instance contrastive learning is sensitive to the data augmentation policies and many trials are needed to determine the right settings [48, 60].

The computational intensity and sensitivity of self-superivsed pretraining may lead researchers to seek self-supervised models from model zoos and research repositories. However, models pretrained on domain-specific datasets are not commonly available. In turn, many practitioners do not use a model pretrained on data similar to their target data, but instead, use a pretrained, publicly available model trained on a large, general dataset, such as ImageNet. We refer to this process as **generalist pretraining**. A growing body of research indicates that pretraining on domain-specific datasets, which we refer to as **specialist pretraining**, leads to improved transfer performance [46, 35, 39].

Figure 1 formalizes this categorization of self-supervised pretraining methods. Generalist and specialist pretraining are as described above, with one round of self-supervised pretraining on a domain-general and domain-specific dataset, respectively. **Hierarchical Pretraining** refers to models pretrained on datasets that are progressively more similar to the target data. HPT first pretrains on a domain-general dataset (referred to as the *base pretrain*), then optionally pretrains on domain-specific datasets (referred to as the *source pretrain*), before finally pretraining on the target dataset (referred to as the *target pretrain*). In all cases, pretraining is followed by supervised finetuning on the target task.

Specialist pretraining presents the same core challenge that transfer learning helps alleviate: a sensitive training process that requires large datasets and significant computational resources [27]. While transfer learning has been carefully investigated in supervised and semi-supervised settings for computer vision [55], it has not been formally studied for self-supervised pretraining, itself. Furthermore, several recent papers that apply self-supervised learning to domain-specific problems did not apply transfer learning to the pretraining process itself, which motivated our work [57, 1, 29].

In this paper, we investigate the HPT framework with a diverse set of pretraining procedures and downstream tasks. We test 16 datasets spanning visual domains, such as medical, aerial, driving, and simulated images. In our empirical study, we observe that HPT shows the following benefits compared to self-supervised pretraining from scratch:

- HPT reduces self-supervised pretraining convergence time up to 80$\times$ compared to pretraining from scratch.
- HPT consistently converges to better performing representations than generalist or specialist pretraining for 15 of the 16 studied datasets on image classification, object detection, and semantic segmentation tasks.
- HPT is significantly more resilient to the set of image augmentations and amount of data used during self-supervised pretraining.

The following sections provide the background, methodology, and experiments used to reach these conclusion and the appendix significantly broadens the scope of our analyses. From this experimental effort, our key conclusion is straightforward: *self-supervised pretraining improves self-supervised pretraining*.

2. Background and Related Work

Transfer learning studies how a larger, more general, or more specialized *source* dataset can be leveraged to improve performance on *target* downstream datasets/tasks [47, 44, 2, 10, 22, 20, 13, 15, 65, 18, 31, 45]. This paper focuses on a common type of transfer learning in which model weights trained on source data are used to initialize training on the target task [63]. Model performance generally scales with source dataset size and the similarity between the source and target data [46, 35, 39].

A fundamental challenge for transfer learning is to improve the performance on target data when it is not similar to source data. Many papers have tried to increase performance when the target and source datasets are not similar. Recently, [43] proposed first training on the base dataset and then training with subsets of the base dataset to create specialist models, and finally using the target data to select the best specialist model. Similarly, [37] used target data to reweight the importance of base data. Unlike these works, we do not revisit the base data, modify the pretrained architecture, or require expert model selection or reweighting.

Self-supervised pretraining is a form of unsupervised training that captures the intrinsic patterns and properties of the data without using human-provided labels to learn discriminative representations for the downstream tasks [11, 12, 66, 17, 58]. In this work we focus on a type of self-supervised pretraining called *instance contrastive learning* [14, 59, 20], which trains a network by determining which visually augmented images originated from the same image, when contrasted with augmented images originating from different images. Instance contrastive learning has recently outperformed supervised pretraining on a variety of transfer tasks [20, 6], which has lead to increased adoption in many applications. Specifically, we use the MoCo algorithm [7] due to its popularity, available code base, reproducible results without multi-TPU core systems, and similarity to other self-supervised algorithms [30]. We also explore additional self-supervised methods in the appendix.

Our focus is on self-supervised learning for vision tasks. Progressive self-supervised pretraining on multiple datasets has also been explored for NLP tasks, e.g. see [19, 42] and the citations within. In [19], the authors compare NLP generalist models with models trained on additional source and task-specific data. While our work is similar in spirit to the language work of [19], our work focuses on computer vision, includes a greater variation of pretraining pipelines, and allows for adaptation with fewer parameter updates.

Label-efficient learning includes weak supervision methods [34] that assume access to imperfect but related labels, and semi-supervised methods that assume labels are only available for a subset of available examples [6, 26, 61]. While some of the evaluations of the learned representations are done in a semi-supervised manner, HPT is complementary to these approaches and the representations learned from HPT can be used in conjunction with them.

3. Hierarchical pretraining

In this section, we formalize each of the HPT components as depicted in Figure 1.

Base pretraining: We use the term *base pretraining* to describe the initial pretraining step where a large, general vision dataset (*base dataset*) is used to pretrain a model from scratch. Practically, few users will need to perform base pretraining, and instead, can use publicly available pretrained models, such as ImageNet models. Because base pretraining, like many prior transfer learning approaches, is domain agnostic, most practitioners will select the highest performing model on a task with a large domain [25].

Source pretraining: Given a base trained model, we select a source dataset that is both larger than the target dataset and more similar to the target dataset than the base dataset. Many existing works have explored techniques to select a model or dataset that is ideal for transfer learning with a target task [49]. Here, we adopt an approach studied by [27, 49] in a supervised context called a *task-aware search strategy*: each potential source dataset is used to perform self-supervised pretraining on top of the base model for a very short amount of pretraining, e.g. $\sim$5k pretraining steps as discussed in Section 4. The supervised target data is then used to train a linear evaluator on the frozen source model. The source model is then taken to be the model that produces the highest linear evaluation score on the target data, and is then used for additional target pretraining.

Experimentally, we have found that using a single, similar, and relatively large (e.g. $>$ 30K images) source dataset consistently improves representations for the target task. Furthermore, we view source pretraining as an optional step, and as shown in Section 4, HPT still leads to improved results when directly performing self-supervised pretraining on the target dataset following the base pretraining. We further discuss source model selection in the appendix.

Target pretraining: Finally, we perform self-supervised pretraining with the target dataset, initialized with the final source model, or the base model in the case when no source model was used. This is also the stage where layers of the model can be frozen to prevent overfitting to the target data and enable faster convergence speed. Experimentally, we have found that freezing all parameters except the modulation parameters of the batch norm layers leads to consistently strong performance for downstream tasks when the target dataset is relatively small ($<$ 10K images).

Supervised finetune: Given the self-supervised pretrained model on the target dataset, we transfer the final model to the downstream target task, e.g. classification.

4. Experiments

Through the following experiments, we investigate the quality, convergence, and robustness of self-supervised pretraining using the HPT framework.

4.1. Datasets

We explored self-supervised pretraining on the following datasets that span several visual domains (see the appendix for all details). Dataset splits are listed with a train/val/test format in square brackets after the dataset description.

Aerial: **xView** [28] is a 36-class object-centric, multi-label aerial imagery dataset [39133/2886/2886]. **RESISC** [8] is a 45-class scene classification dataset for remote sensing [18900/6300/6300]. **UC-Merced** [62] is a 21-class aerial imagery dataset [1260/420/420].

Autonomous Driving: **BDD** [64] is a high resolution driving dataset with 10 object detection labels and 6 weather classification labels. We evaluate HPT performance over the object detection task, as well as the weather classification task [$60k/10k/10k$]. **VIPER** [50] is a 23-class simulated driving dataset for which we perform multi-label each object in the image [13367/2868/4959].

Medical: **Chexpert** [23] is a large, multi-label X-ray dataset, where we determine whether each image has any of 5 conditions [178731/44683/234]. **Chest-X-ray-kids** [24] provides pediatric X-rays used for 4-way pneumonia classification [4186/1046/624].

Natural, Multi-object: **COCO-2014** [32] is an 81-class object detection benchmark. We perform multi-label classification for each object, and we further use the 2017 split to perform object detection and segmentation [82783/20252/20252]. **Pascal** VOC 2007+2012 [16] is a standard 21-class object detection benchmark we use for multi-label classification to predict whether each object is in each image. We also use the object detection labels for an object detection transfer task [$13.2k/3.3k/4.9k$].

Assorted: **DomainNet** [41] contains six distinct datasets, where each contains the same 345 categories. The domains consist of `real` images similar to ImageNet, `sketch` images of greyscale sketches, `painting` images, `clipart` images, `quickdraw` images of binary black-and-white drawings from internet users, and `infograph` illustrations. We use the original train/test splits with 20% of the training data used for validation. **Oxford Flowers** [38]: we use the standard split to classify 102 fine-grain flower categories [1020/1020/6149].

4.2. Evaluations

The features of self-supervised pretrained models are typically evaluated using one of the following criteria:

- **Separability**: Tests if a linear model can differentiate classes in a dataset using learned features. Good representations should be linearly separable [40, 9].

- **Transferability**: Tests the performance of the model

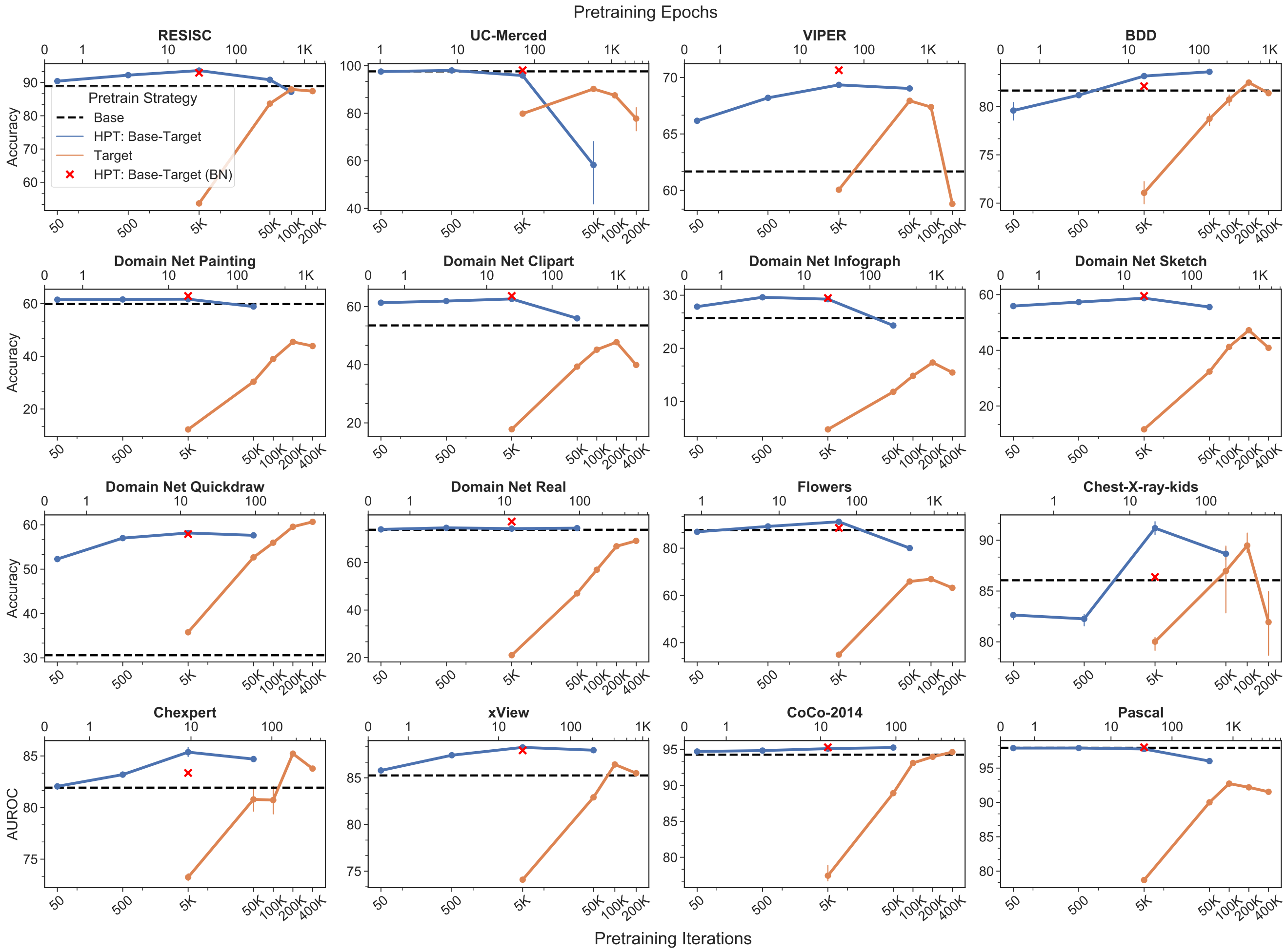

Figure 2. *Linear separability evaluation.* For each of the 16 datasets, we train a generalist model for 800 epochs on ImageNet (Base). We either train the whole model from 50-50k iters (HPT Base-Target) or just the batch norm parameters for 5k iters (HPT Base-Target (BN)). We compare HPT to a Specialist model trained from a random initialization (Target). For each, we train a linear layer on top of the final representation. HPT obtains the best results on 15 out of 16 datasets without hyperparameter tuning.

when finetuned on new datasets and tasks. Better representations will generalize to more tasks [20].

- **Semi-supervised**: Test performance with limited labels. Better representations will suffer less performance degradation [22, 5].

We explored these evaluation methods with each of the above datasets. For all evaluations, unless otherwise noted, we used a single, centered crop of the test data with no test-time augmentations. For classification tasks, we used top-1 accuracy and for multi-label classification tasks we used the Area Under the ROC (AUROC) [3].

In our experiments, we used MoCo-V2 [7] as the self-supervised training algorithm. We selected MoCo-V2 as it has state-of-the-art or comparable performance for many transfer tasks, and because it uses the InfoNCE loss function [40], which is at the core of many recent contrastive pretraining algorithms [33]. Unless otherwise noted, all training is performed with a standard ResNet-50 backbone [54] on 4 GPUs, using default training parameters from [20]. We also explored additional self-supervised pretraining algorithms and hyperparameters in the appendix.

In the following experiments, we compare implementations of the following self-supervised pretraining strategies:

- **Base**: transfers the 800-epoch MoCo-V2 ImageNet model from [7] and also updates the batch norm's non-trainable mean and variance parameters using the target dataset (this uniformly led to slightly improved performance for Base transfer).

- **Target**: performs MoCo-V2 on the target dataset from scratch.

- **HPT**: initializes MoCo-V2 pretraining with the 800-epoch MoCo-V2 ImageNet model from [7], then op-

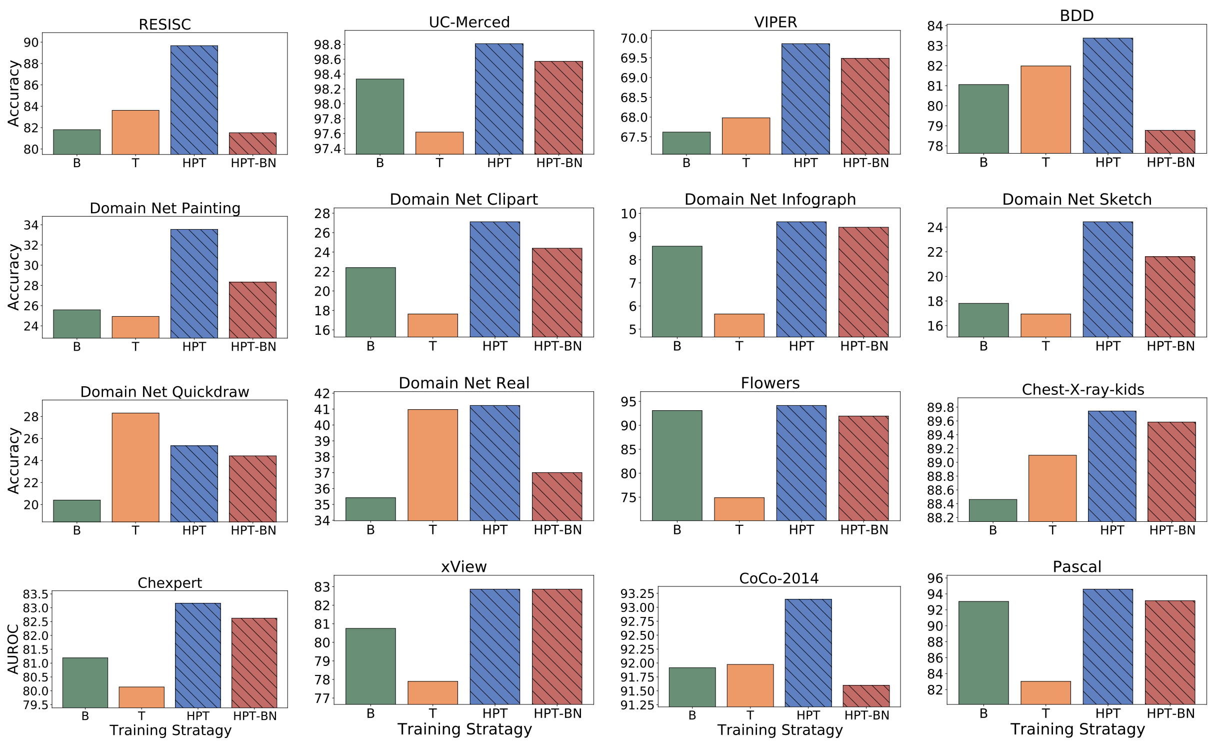

Figure 3. *Semi-supervised evaluation*. We compared the best semi-supervised finetuning performance from the (B)ase model, (T)arget pretrained model, HPT pretrained model, and HPT-BN pretrained model using a 1k labeled subset of each dataset. Despite performing 10x-80x less pretraining, HPT consistently outperformed the Base and Target. HPT-BN generally showed improvement over Base model transfer, but did not surpass HPT's performance.

tionally performs pretraining on a source dataset before pretraining on the target dataset. The batch norm variant (**HPT-BN**) only trains the batch norm parameters (γ, β), e.g. a ResNet-50 has 25.6M parameters, where only $\sim$0.2% are BN parameters.

Existing work largely relies on supervised evaluations to tune the pretraining hyperparameters [5], but in practice, it is not possible to use supervised evaluations of unlabeled data to tune the hyperparameters. Therefore, to emphasize the practicality of HPT, we used the default pretraining hyperparameters from [7] with a batch size of 256 (see the appendix for full details).

4.3. Pretraining Quality Analysis

Separability analysis: We first analyzed the quality of the learned representations through a linear separability evaluation [5]. We trained the linear model with a batch size of 512 and the highest performing learning rate of $\{0.3, 3, 30\}$. Similar to [27], we used steps rather than epochs to allow for direct computational comparison across datasets. For Target pretraining, we pretrained for {5k, 50k, 100k, 200k, 400k} steps, where we only performed 400k steps if there was an improvement between 100k and 200k steps. For reference, one NVIDIA P100 GPU-Day is 25k steps. We pretrained HPT for much shorter schedules of {50, 500, 5k, 50k} steps, and HPT-BN for 5k steps – we observed little change for HPT-BN after 5k steps.

Key observations: From Figure 2, we observe that HPT typically converges by 5k steps of pretraining *regardless of the target dataset size*, and that for 15 out of 16 datasets, HPT and HPT-BN converged to models that performed as well or better than the Base transfer or Target pretraining at 400k steps (80x longer). The only dataset in which the Target pretraining outperformed HPT was `quickdraw` – a large, binary image dataset of crowd-sourced drawings. We note that `quickdraw` is the only dataset in Target pretraining at 5k steps outperformed directly transferring the Base model, indicating that the direct transfer performance from ImageNet is quite poor due to a large domain gap – an observation further supported by its relatively poor domain adaptation in [41].

HPT improved performance on RESISC, VIPER, BDD, Flowers, xView, and `clipart`, `infograph`, and `sketch`: a diverse range of image domains and types. HPT had similar performance as Base transfer for the datasets that were most similar to ImageNet: `real`, COCO-2014, and Pascal, as well as for UC-Merced, which had 98.2% accuracy for Base transfer and 99.0% accuracy for HPT and

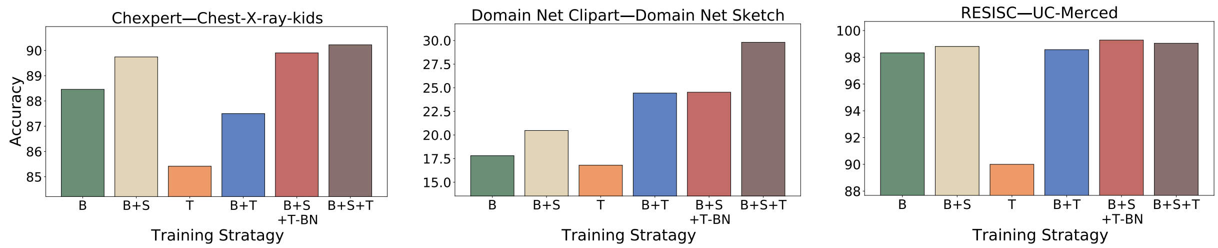

Figure 4. *Full finetuning evaluations.* Finetuning performance on target datasets. For these datasets, we evaluated the performance increase on the target dataset by pretraining on sequences of (B)ase (ImageNet), (S)ource (left) dataset, and (T)arget (right) dataset. All HPT variants beat all baselines, with HPT-BN getting slightly better performance on UC Merced and B+S+T having the best performance elsewhere.

HPT-BN. The two medical datasets, Chexpert and Chest-X-ray-kids had comparable performance with HPT and Target pretraining, yet HPT reached equivalent performance in 5k steps compared to 200k and 100k, respectively. Finally, HPT exhibited overfitting characteristics after 5k steps, where the overfitting was more pronounced on the smaller datasets (UC-Merced, Flowers, Chest-X-ray-kids, Pascal), leading us to recommend a very short HPT pretraining schedule, e.g. 5k iterations, regardless of dataset size (see appendix for additional experiments).

Semi-supervised transferability: Next, we conducted a semi-supervised transferability evaluation of the pretrained models. This experiment tested whether the benefit from the additional pretraining is nullified when finetuning all model parameters. Specifically, we selected the top performing models from the linear analysis for each pretraining strategy and fully finetuned the pretrained models using 1000 randomly selected labels without class balance but such that each class occured at least once. We finetune using a combination of two learning rates (0.01, 0.001) and two finetuning schedules (2500 steps, 90 epochs) with a batch size of 512 and report the top result for each dataset and model – see the appendix for all details.

Key observations: Figure 3 shows the top finetuning performance for each pretraining strategy. The striped bars show the HPT pretraining variants, and we observe that similar to the linear analysis, HPT has the best performing pretrained models on 15 out of 16 datasets, with `quickdraw` being the exception. One key observation from this experiment is that HPT is beneficial in the semi-supervised settings and that the representational differences from HPT and the Base model are different enough that full model finetuning cannot account for the change. We further note that while HPT-BN outperformed HPT in several linear analyses, HPT-BN never outperformed HPT when finetuning all parameters. This result indicates that some of the benefit from pretraining only the batch norm parameters is redundant with supervised finetuning. We also note that whether Base or Target pretraining performed better depended on the dataset, while HPT had uniformly strong performance.

Sequential pretraining transferability: Here, we explore HPT's performance when pretraining on a source dataset before pretraining on the target dataset and finally transferring to the target task. We examined three diverse target datasets: Chest-X-ray-kids, `sketch`, and UC-Merced. We select the source dataset for each of the target dataset by choosing the source dataset that yielded the highest linear evaluation accuracy on the target dataset after 5k pretraining steps on top of the base model. This selection yielded: ImageNet then Chexpert then Chest-X-ray-kids, ImageNet then `clipart` then `sketch`, and ImageNet then RESISC then UC-Merced.

Key observations: Figure 4 compares finetuning the 1000-label subset of the target data after the following pretraining strategies: directly using the Base model (B), Target pretraining (T), Base then Source pretraining (B+S), Base then Target pretraining (B+T), Base then Source pretraining then Target pretraining (B+S+T), and Base then Source pretraining then Target pretraining on the batch norm parameters (B+S+T-BN). The full HPT pipeline (B+S+T) leads to the top results on all three target datasets. In the appendix, we further show that the impact of an intermediate source model decreases with the size of the target.

Object detection and segmentation transferability: For Pascal and BDD, we transferred HPT pretrained models to a Faster R-CNN R50-C4 model and finetuned the full model; for COCO, we used a Mask-RCNN-C4. Over three runs, we report the median results using the COCO AP metric as well as AP_{50}/AP_{75}. For Pascal, we performed finetuning on the `train2007+2012` set and performed evaluation on the `test2007` set. For BDD we used the provided train/test split, with 10k random images in the train split used for validation. For COCO, we used the 2017 splits and trained with the 1x schedule (see appendix for all details).

Key observations: Tables 1-2 show the object detection and segmentation results. For Pascal, we tested HPT instantiations of Base-Target, Base-Target (BN), and Base-Source-Target, where COCO-2014 was selected as the source model using the top-linear-analysis selection criteria. For the larger BDD and COCO datasets, we tested Base-Target and Base-Target (BN). Overall, the results are consistent across all datasets for image classification, ob-

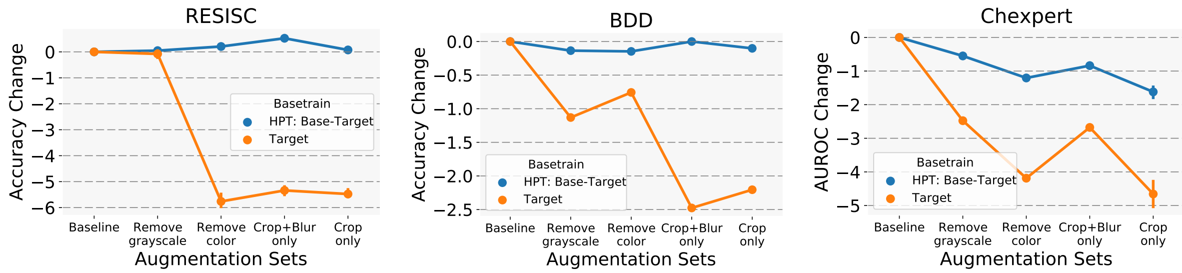

Figure 5. *Augmentation robustness*. We compare the accuracy change of sequentially removing data augmentations on linear evaluation performance. HPT performs better with only cropping than any other policy does with any incomplete combination.

Table 1. Transfer Result: This table reports the median AP, AP_{50}, AP_{75} over three runs of finetuning a Faster-RCNN C4 detector. For Pascal, the Source dataset is COCO-2014. A bold result indicates a $+0.2$ improvement over all other pretraining strategies.

Pretrain	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
Pascal VOC07			
Target	48.4	75.9	51.9
Base	57.0	82.5	63.6
HPT: Base-Target	57.1	**82.7**	63.7
HPT: Base-Target (BN)	**57.5**	**82.8**	64.0
HPT: Base-Source-Target	**57.5**	**82.7**	**64.4**
HPT: Base-Source-Target (BN)	**57.6**	**82.9**	64.2
BDD			
Target	24.3	46.9	24.0
Base	27.1	48.7	25.4
HPT: Base-Target	**28.1**	**50.0**	**26.3**
HPT: Base-Target (BN)	**28.0**	49.6	**26.3**

ject detection, and segmentation: HPT: both Base-Target and Base-Target (BN) lead to improvements over directly transferring the Base model to the target task.

The Base-Source-Target Pascal results show an improvement when pretraining all model parameters, but remain consistent when only pretraining the batch norm parameters. This indicates that while the batch norm parameters can improve the pretrained model, sequentially pretraining from the source to the target on these values does not always yield an improved result. While the overall gains are modest, these results indicate that HPT is not directly learning redundant information with either the MoCo pretraining or the finetuning task. Furthermore, it is surprising that only tuning the batch norm parameters on the target dataset leads to an improvement in object detection, and we note that pretraining specific subsets of object detector backbone parameters may provide a promising direction for future work.

4.4. HPT Robustness

Here, we investigate the robustness of HPT to common factors that impact the effectiveness of self-supervised pretraining such as the augmentation policy [5, 48] and pretraining dataset size [36]. For these robustness experiments, we used the BDD, RESISC, and Chexpert datasets as they provided a diversity in data domain and size.

Table 2. Transfer Result: This table reports the median AP, AP_{50}, AP_{75} over three runs of finetuning a Mask-RCNN-C4 detector on COCO-2017. A bold result indicates at least a 0.2 improvement over all other pretraining strategies.

Pretrain	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
Target	36.0	54.7	38.6	19.3	40.6	49.1
Base	38.0	57.4	41.3	20.7	43.3	51.4
HPT: B-T	**38.4**	**58.0**	41.3	**21.6**	**43.5**	**52.2**
HPT: B-T (BN)	38.2	57.4	40.9	20.6	43.4	**52.2**

Augmentation robustness: MoCo-V2 sequentially applies the following image augmentations: `RandomResizedCrop`, `ColorJitter`, `Grayscale`, `GaussianBlur`, `RandomHorizontalFlip`. We studied the robustness of HPT by systematically removing these augmentations and evaluating the change in the linear evaluation for HPT and Target pretraining.

Key observations: Figure 5 shows separability results across datasets after sequentially removing augmentations. In all three data domains, HPT maintained strong performance compared to Target pretraining. Unlike BDD and RESISC, the Chexpert performance decreased as the augmentation policy changed. This illustrates that changes to the augmentation policy can still impact performance when using HPT, but that the overall performance is more robust. In turn, as a practitioner explores a new data domain or application, they can either use default augmentations directly or choose a conservative set, e.g. only cropping.

Pretraining data robustness: We pretrained with $\{1\%, 10\%, 25\%, 100\%\}$ of the target dataset. For HPT we used 5k pretraining steps. With 25% or 100% of the data, we used the same number of steps as the top performing result in Figure 2, and $1/10$ of the steps at 1% and 10%.

Key observations: Figure 6 shows separability results. CheXpert has 3x more training data than BDD, which in turn has 3x more training data than RESISC. While more data always performed better, the accuracy improvements of HPT increased as the amount of pretraining data decreased. HPT-BN had minimal degradation in low data regimes – outperforming other methods with <5k samples.

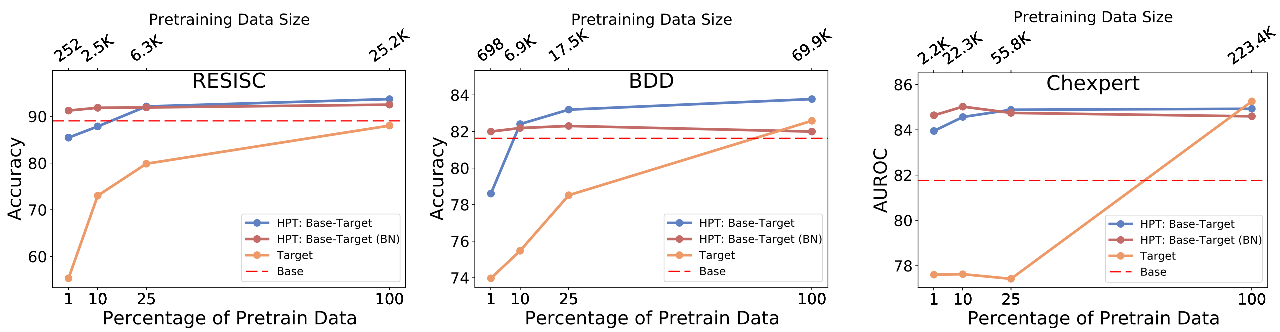

Figure 6. *HPT performance as the amount of pretraining data decreases*. The top axis shows the number of images, and the bottom shows the percentage of pretraining data. HPT outperforms Base model transfer or Target pretraining with limited data.

Table 3. Budget levels and test accuracy in target domain for semi-supervised DA with and without HPT between `real`→`clip` and `real`→`sketch`.

	Budget levels in target domains						
# of shots	1	11	16	22	32	46	68
	Test accuracy (%) for `real`→`clip`						
MME	49.7	61.1	63.9	66.7	68.0	70.0	71.1
MME+HPT	**57.2**	**64.36**	**66.7**	**68.2**	**69.7**	**71.5**	**72.4**
	Test accuracy (%) for `real`→`sketch`						
MME	41.4	51.8	54.9	57.5	59.7	61.4	62.5
MME+HPT	**50.2**	**56.4**	**58.8**	**60.7**	**62.8**	**63.9**	**64.9**

4.5. Domain Adaptation Case Study

In this section, we explore the utility of HPT through a realistic case study experiment in which we apply HPT in a domain adaptation context. The training procedure is as follows: we performed HPT to train a model using both source and target datasets on top of the standard MSRA ImageNet model [21]. We used this model to initialize the feature encoder in a Minimax Entropy (MME) Domain Adaptation (DA) method [52]. At the end of each budget level we evaluated accuracy on the entire test set from the target domain. We perform two experiments on DomainNet datasets [41] with 345 classes in 7 budget levels with increasing amount of target labels: (i) from `real` to `clip` and (ii) from `real` to `sketch` and use a `EfficientNet_B2` [56] backbone.

Table 3 shows the benefit of adding HPT to the DA experiments. From the results, we observe that HPT consistently outperforms the baseline on both domains by achieving a higher accuracy across all the budget levels. On the extreme low data regime (one shot/class), HPT achieves nearly 8% better accuracy in both `clipart` and `sketch` domains in the extreme case of providing one shot per class in the target domain. These results demonstrate HPT's effectiveness in a realistic, end-to-end inference system.

5. Discussion

We have shown that HPT achieves faster convergence, improved performance, and increased robustness across domains. Here, we further reflect on the utility of the HPT.

What is novel about HPT? The transfer learning methodology underlying HPT is well established in transfer learning. That is, transfer learning tends to work in a lot of situations, and our work could be perceived as a natural extension of this general observation. However, our work provides the first thorough empirical analysis of transfer learning applied to self-supervised pretraining in computer vision. We hope this analysis encourages practitioners to include an HPT baseline in their investigations – a baseline that is surprisingly absent from current works.

How should I use HPT in practice? We provide our code, documentation, and models to use HPT and reproduce our results (see appendix). For existing codebases, using HPT is usually as simple as starting training from an existing checkpoint. If working with a small dataset (e.g. $<$ 10k images), using HPT-BN is ideal.

Does this work for supervised learning? Yes. In the appendix, we reproduce many of these analyses using supervised ImageNet base models and show that HPT further improves performance across datasets and tasks.

6. Conclusion and Implications

Our work provides the first empirical analysis of transfer learning applied to self-supervised pretraining for computer vision tasks. In our experiments, we have observed that HPT resulted in 80x faster convergence, improved accuracy, and increased robustness for the pretraining process. These results hold across data domains, including aerial, medical, autonomous driving, and simulation. Critically HPT requires fewer data and computational resources than prior methods, enabling wider adoption of self-supervised pretraining for real-world applications. Pragmatically, our results are easy to implement and use: we achieved strong results without optimizing hyperparameters or augmentation policies for each dataset. Taken together, HPT is a simple framework that improves self-supervised pretraining while decreasing resource requirements.

References

[1] Kumar Ayush, Burak Uzkent, Chenlin Meng, Marshall Burke, David Lobell, and Stefano Ermon. Geography-aware self-supervised learning. *arXiv preprint arXiv:2011.09980*, 2020.

[2] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings, 2012.

[3] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv:2002.05709 [cs, stat]*, Mar. 2020. arXiv: 2002.05709.

[6] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020.

[7] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning. *arXiv:2003.04297 [cs]*, Mar. 2020. arXiv: 2003.04297.

[8] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

[9] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural networks: Tricks of the trade*, pages 561–580. Springer, 2012.

[10] 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.

[11] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[12] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

[13] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[14] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1734–1747, 2015.

[15] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

[16] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[17] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations, 2018.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[19] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[21] 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.

[22] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

[23] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 590–597, 2019.

[24] Daniel Kermany, Kang Zhang, and Michael Goldbaum. Large dataset of labeled optical coherence tomography (oct) and chest x-ray images. *Mendeley Data, v3 http://dx. doi. org/10.17632/rscbjbr9sj*, 3, 2018.

[25] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*, 6(2):8, 2019.

[26] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning, 2020.

[27] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2661–2671, 2019.

[28] Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xview: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018.

[29] Nick Lamm, Shashank Jaiprakash, Malavika Srikanth, and Iddo Drori. Vehicle trajectory prediction by trans-

fer learning of semi-supervised models. *arXiv preprint arXiv:2007.06781*, 2020.

[30] Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 2020.

[31] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[32] 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.

[33] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 1(2), 2020.

[34] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

[35] Maxim Neumann, Andre Susano Pinto, Xiaohua Zhai, and Neil Houlsby. In-domain representation learning for remote sensing. *arXiv preprint arXiv:1911.06721*, 2019.

[36] Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7345–7354, 2020.

[37] Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V Le, and Ruoming Pang. Domain adaptive transfer learning with specialist models. *arXiv preprint arXiv:1811.07056*, 2018.

[38] 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.

[39] Shuteng Niu, Meryl Liu, Yongxin Liu, Jian Wang, and Houbing Song. Distant domain transfer learning for medical imaging. *arXiv preprint arXiv:2012.06346*, 2020.

[40] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[41] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[42] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, 2020.

[43] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, Cedric Renggli, André Susano Pinto, Sylvain Gelly, Daniel Keysers, and Neil Houlsby. Scalable transfer learning with expert models. *arXiv preprint arXiv:2009.13239*, 2020.

[44] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Transfer learning for image classification with sparse prototype representations. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[45] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

[46] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. *arXiv preprint arXiv:1902.07208*, 2019.

[47] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766, 2007.

[48] Colorado J Reed, Sean Metzger, Aravind Srinivas, Trevor Darrell, and Kurt Keutzer. Selfaugment: Automatic augmentation policies for self-supervised learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2021.

[49] Cedric Renggli, André Susano Pinto, Luka Rimanic, Joan Puigcerver, Carlos Riquelme, Ce Zhang, and Mario Lucic. Which model to transfer? finding the needle in the growing haystack. *arXiv preprint arXiv:2010.06402*, 2020.

[50] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2213–2222, 2017.

[51] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[52] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8050–8058, 2019.

[53] H. Shin, M. Orton, D. J. Collins, S. Doran, and M. O. Leach. Autoencoder in time-series analysis for unsupervised tissues characterisation in a large unlabelled medical image dataset. In *2011 10th International Conference on Machine Learning and Applications and Workshops*, volume 1, pages 259–264, 2011.

[54] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[55] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pages 270–279. Springer, 2018.

[56] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[57] Chao Tao, Ji Qi, Weipeng Lu, Hao Wang, and Haifeng Li. Remote sensing image scene classification with self-supervised paradigm under limited labeled samples. *IEEE Geoscience and Remote Sensing Letters*, 2020.

[58] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matthew J Kusner. Pre-training by completing point clouds. *arXiv preprint arXiv:2010.01089*, 2020.

[59] 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.

[60] Tete Xiao, Xiaolong Wang, Alexei A Efros, and Trevor Darrell. What should not be contrastive in contrastive learning. *arXiv preprint arXiv:2008.05659*, 2020.

[61] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*, 2019.

[62] 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, 2010.

[63] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.

[64] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[65] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

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