Streaming Self-Training via Domain-Agnostic Unlabeled Images

Zhiqiu Lin   Deva Ramanan   Aayush Bansal

Carnegie Mellon University

https://www.cs.cmu.edu/~aayushb/SST/

Abstract

We present streaming self-training (SST) that aims to democratize the process of learning visual recognition models such that a non-expert user can define a new task depending on their needs via a few labeled examples and minimal domain knowledge. Key to SST are two crucial observations: (1) domain-agnostic unlabeled images enable us to learn better models with a few labeled examples without any additional knowledge or supervision; and (2) learning is a continuous process and can be done by constructing a schedule of learning updates that iterates between pretraining on novel segments of the streams of unlabeled data, and fine-tuning on the small and fixed labeled dataset. This allows SST to overcome the need for a large number of domain-specific labeled and unlabeled examples, exorbitant computational resources, and domain/task-specific knowledge. In this setting, classical semi-supervised approaches require a large amount of domain-specific labeled and unlabeled examples, immense resources to process data, and expert knowledge of a particular task. Due to these reasons, semi-supervised learning has been restricted to a few places that can house required computational and human resources. In this work, we overcome these challenges and demonstrate our findings for a wide range of visual recognition tasks including fine-grained image classification, surface normal estimation, and semantic segmentation. We also demonstrate our findings for diverse domains including medical, satellite, and agricultural imagery, where there does not exist a large amount of labeled or unlabeled data.

1. Introduction

Our goal is to democratize the process of learning visual recognition models for a non-expert[1] user who can define a new task (as per their needs) via a few labeled examples and minimal domain knowledge. A farmer may want to create a visual recognition model to protect their seasonal crops

---

[1]A non-expert is someone whose daily job is not to train visual recognition models but is interested in using them for a specific application according to their personal needs.

(a) Children continually improve their knowledge about a concept.

Flowers-102 dataset: 10 examples per class
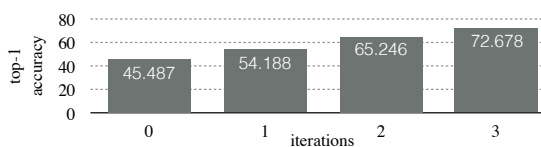
(b) Machines can also improve their knowledge about a concept in this iterative manner.

Figure 1. **(a)** We take inspiration from developmental psychology that explores how children learn. Children maybe exposed to a concept (say flowers), play with other things in their environment, and eventually return to the lesson at hand. By interleaving periods of self-supervised play and teacher-supervised learning, they continuously improve their performance. **(b)** We take inspiration from this model of iterative and streaming learning, and show how machines can learn better representations for various visual understanding tasks. Here, we show an illustrative fine-grained task of classifying flowers [53]. *Iteration*-0 shows the performance of a convolutional neural network trained from scratch using only 10 examples per class. We use streams of unlabeled images as a proxy of exploring the world. We improve the performance as we progress along the stream. At the end of the third iteration, the performance on the task improved from 45.49% to 72.68% top-1 accuracy. Note this is done without any other labeled data, without any modifications for the task or domain (fine-grained classification), and without any tuning of hyperparameters.

and vegetables from diseases or insects. A conservatory may want to segment their flowers and butterflies. A sanctuary may want to identify their migratory birds. A meteorologist may want to study satellite images to understand monsoonal behavior in a region. Currently, training a visual recognition model requires enormous domain knowledge, specifically (1) large amount of domain-specific labeled or unlabeled data; (2) extensive computational resources (disk space to store
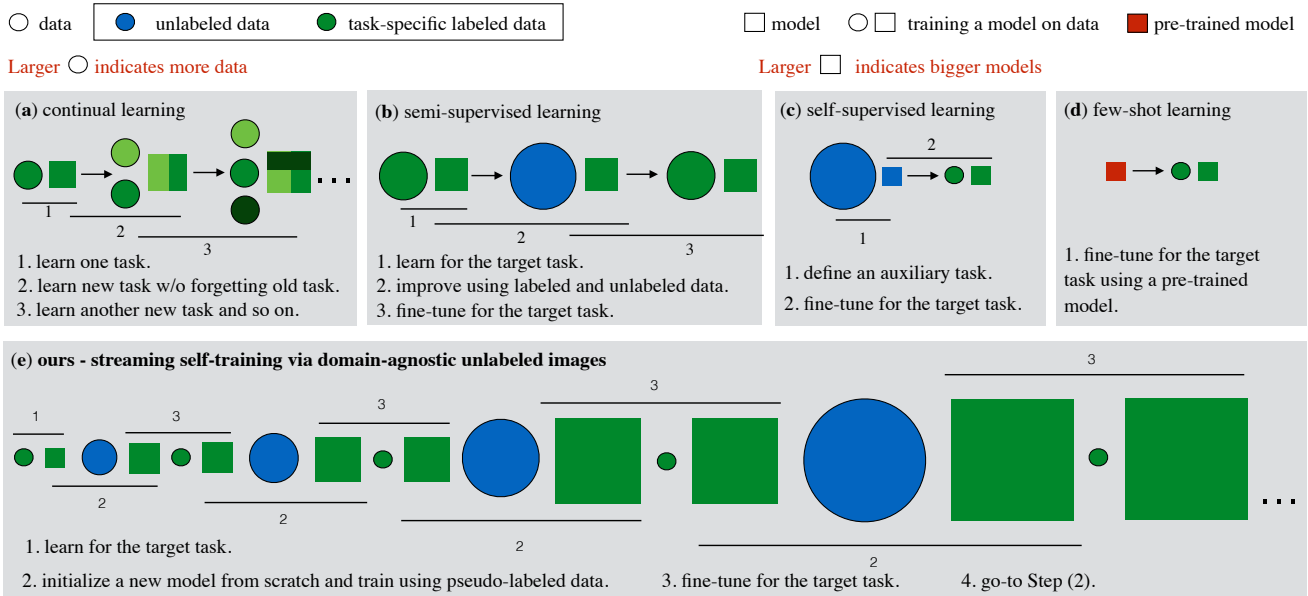
Figure 2. Contrasting Streaming Self-Training (SST) with Established Methods: (a) **Continual learning** continually learns new tasks in a supervised manner without forgetting previous ones. SST continuously learn better models for a fixed task using an infinite stream of unlabeled data.(b) **Semi-supervised learning** typically requires (1) a large domain-specific unlabeled dataset sampled from same or similar data distribution as that of labeled examples [9, 10, 15, 35, 43, 55, 70, 90]; (2) intensive computational resources [58, 91, 93]; and (3) task-specific knowledge such as better loss-functions for image classification tasks [5, 8] or cleaning noisy pseudo-labels [4, 35, 41, 91, 93]. In contrast, SST makes use of unlabeled data that is domain-agnostic and has no relation with the intended task. SST also requires modest compute; we use a 4 GPU (GeForce RTX 2080) machine to conduct all our experiments. (c) **Self-supervised learning** learns a generic task-agnostic representation from unlabeled images, which may struggle when applied to data distributions that differ from the unlabeled data [21, 51, 80]. On the contrary, SST learns better models for a task via unlabeled images from drastically different data distribution. Our work is closely related to the recent work [14] that use big self-supervised models for semi-supervised learning. We observe that same insights hold even when using impoverished models for initialization, i.e., training the model from scratch for a task given a few labeled examples. The performance for the task is improved over time in a streaming/iterative manner. While we do observe the benefits of having a better initialization, we initialize the models from scratch for a task throughout this work. (d) **Few-shot learning** learns representations from a few-labeled examples. Guo et al. [28] show that popular few-shot learning methods [22, 42, 69, 72, 77, 79] underperform simple finetuning, i.e., when a model pre-trained on large annotated datasets from similar domains is used as an initialization to the few-shot target task. The subsequent tasks in few-shot learners are often tied to both original data distribution and tasks. SST makes use of few-labeled examples but it is both task-agnostic and domain-agnostic.

data and GPUs to process it); (3) task-specific optimization or dataset-specific knowledge to tune hyperparameters. As researchers, we take these things for granted. It is non-trivial for a non-expert to collect a large amount of task-specific labeled data, have access to industry-size computational resources, and certainly not have the expertise with tasks and tuning hyperparameters. Even recent semi-supervised approaches [91, 93] (not requiring extensive labeled data) may cost a million dollar budget for AWS compute resources.

As a modest step toward the grand goal of truly democratic ML, we present streaming self-training (SST), which allows users to learn from a few labeled examples and a domain-agnostic unlabeled data stream. SST learns iteratively on chunks of unlabeled data that overcomes the need for storing and processing large amounts of data. Crucially, SST can be applied to a wide variety of tasks and domains

without task-specific or domain-specific assumptions. A non-expert can get better models that continuously improve by self-training on a universal stream of unlabeled images that are agnostic to the task and domain. SST is loosely inspired by theories of cognitive development (Fig. 1), whereby children are able to learn a concept (apple, banana, etc) from a few labeled examples and continuous self-play without explicit teacher feedback [24].

Self-Training and Semi-Supervised Learning: A large variety of self-training [19, 85] and semi-supervised approaches [58, 65, 78, 94, 91, 93] use unlabeled images in conjunction with labeled images to learn a better representation (Fig. 2-(b)). These approaches require: (1) a large domain-specific unlabeled dataset sampled from same or similar data distribution as that of labeled examples [9, 10, 15, 35, 43, 55, 70, 90]; (2) intensive computational require-
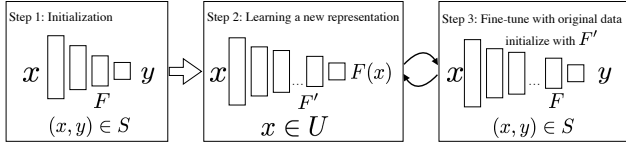
Figure 3. **Our Approach:** There are three important steps of our approach. (a) **Step 1:** Initialization– we learn an initial mapping F on $(x, y) \in S$; (b) **Step 2:** Learning a new representation– We use F to learn a new model F' from scratch on sample $x \in U$; and (c) finally, **Step 3:** Fine-tune with original data – we fine-tune F' on S. This becomes our new F. We continually cycle between Step-2 and Step-3. The capacity of model F' increases with every cycle.

ments [58, 91, 93]; and (3) task-specific knowledge such as better loss-functions for image classification tasks [5, 8] or cleaning noisy pseudo-labels [4, 35, 41, 91, 93]. We differ from this setup. In this work, the unlabeled data is domain-agnostic and have no relation with the intended task. We use a 4 GPU (GeForce RTX 2080) machine to conduct all our experiments. Finally, we do not apply any advanced optimization schema, neither we apply any task-specific knowledge nor we tune any hyperparameters. In this work, we emulate the settings of a non-expert user as best as possible.

Domain-Agnostic Unlabeled Streams: A prevailing wisdom is that unlabeled data should come from relevant distributions [9, 10, 15, 35, 43, 55, 70, 90]. In this work, we make the somewhat surprising observation that unlabeled examples from quite different data distributions can still be helpful. We make use of an universal, unlabeled stream of web images to improve a variety of domain-specific tasks defined on satellite images, agricultural images, and even medical images. Starting from a few labeled examples, we iteratively improve task performance by constructing a schedule of learning updates that iterates between pre-training on segments of the unlabeled stream and fine-tuning on the small labeled dataset (Fig. 2-(e)). We progressively learn more accurate pseudo-labels as the stream is processed. This observation implies that we can learn better mappings using diverse unlabeled examples without any extra supervision or knowledge of the task.

Our Contributions: (1) We study the role of domain-agnostic unlabeled images to learn a better representation for a wide variety of tasks without any additional assumption and auxiliary information. We demonstrate this behaviour for the tasks where data distribution of unlabeled images drastically varies from the labeled examples of the intended task. A simple method utilizing unlabeled images allows us to improve performance of medical-image classification, crop-disease classification, and satellite-image classification. Our insights (without any modification) also hold for pixel-level prediction problems. We improve surface normal estimation on NYU-v2 depth dataset [66] and semantic segmentation on PASCAL VOC-2012 [20] by $3 - 7\%$; (2) We then

demonstrate that one can continuously improve the performance by leveraging more streams of unlabeled data. Since we have potentially an infinite streams of unlabeled data, we can continuously learn better task-specific representations. We specifically demonstrate it for fine-grained image classification tasks. Without adding any domain-specific or task-specific knowledge, we improve the results in few iterations of our approach. We also demonstrate that our approach enables to train very high capacity models on a few-labeled example per class with minimal knowledge of neural networks; and (3) finally, we study that how these insights allow us to design an efficient and cost-effective system for a non-expert.

2. Related Work

SST is inspired from the continuously improving and expanding human mind [2, 3]. Prior work focuses on one-stage approaches for learning representations for a task, typically via more labeled data [46, 64, 98], higher capacity parametric models [31, 33, 40, 68], finding better architectures [11, 73, 101], or adding task-specific expert knowledge to train better models [56, 83].

Continual and Iterated Learning: Our work shares inspiration with a large body of work on continual and life-long learning [74, 75, 67]. A major goal in this line of work [22, 23, 60, 62, 81] has been to continually learn a good representation over a sequence of tasks (Fig. 2-(a)) that can be used to adapt to a new task with few-labeled examples without forgetting the earlier tasks [12, 45]. Our goal, however, is to learn better models for a task given a few labeled examples without any extra knowledge. Our work shares insights with iterated learning [38, 39] that suggests evolution of language and emerging compositional structure of human language through the successive re-learning. Recent work [48, 49] has also used these insights in countering language drift and interactive language learning. In this work, we restrict ourselves to visual recognition tasks and show that we can get better task performance in an iterated learning fashion using infinite stream of unlabeled data.

Learning from Unlabeled or Weakly-Labeled Data: The power of large corpus of unlabeled or weakly-labeled data has been widely explored in semi-supervised learning [4, 13, 35, 52, 57, 58, 59, 97, 100], self-supervised learning (Fig. 2-(c)) [18, 26, 96], or weakly-supervised learning [36, 37, 71, 99]. While self-supervised approaches aim to learn a generic task-agnostic representation from unlabeled images, they may struggle when applied to data distributions that differ from the unlabeled data [21, 51, 80]. On the contrary, SST learns better models for a task via unlabeled images from drastically different data distribution. A wide variety of work in few-shot learning [44, 61, 84, 88], meta-learning [63, 69, 72] aims to learn from few labeled samples. These approaches largely aim at learning a better generic

visual representation from a few labeled examples (Fig. 2-(d)). In this work, we too use few labeled samples for the task of interest along with large amounts of domain-agnostic unlabeled images. Our goal is to learn a better model for any task without any domain biases, neither employing extensive computational resources nor expert human resources. Our work is closely related to the recent work [14] that use big self-supervised models for semi-supervised learning. We observe that same insights hold even when using impoverished models for initialization, i.e., training the model from scratch for a task given a few labeled examples. The performance for the task is improved over time in a streaming/iterative manner. While we do observe the benefits of having a better initialization (Sec 4.1.3), we initialize the models from scratch for a task for all our analysis throughout this work.

Domain Biases and Agnosticism: Guo et al. [28] show that meta-learning methods [22, 42, 69, 72, 77, 79] underperform simple finetuning, i.e., when a model pre-trained on large annotated datasets from similar domains is used as an initialization to the few-shot target task. The subsequent tasks in few-shot learners are often tied to both original data distribution and tasks. SST makes use of few-labeled examples but it is both task-agnostic and domain-agnostic. In this work, we initialize models from scratch (random gaussian initialization) from a few labeled examples. In many cases, we observe that training from scratch with a few-labeled examples already competes with fine-tuning a model pretrained on large labeled dataset (e.g., medical and satellite image classification, and surface normal estimation). Our work is both domain- and task-agnostic. We show substantial performance improvement in surface normal estimation [25, 83] on NYU-v2-depth [66] (that is primarily an indoor world dataset collected using a Kinect) via an unlabeled stream of web images. We similarly show that unlabeled internet streams can be used to improve classification accuracy of crop-diseases [64], satellite imagery [32], and medical images [16, 76] with even a modest number of labeled examples (20 examples per class).

Avoiding Overfitting: An important consequence of our work is that we can now train very deep models from scratch using a few labeled examples without any expert neural network knowledge. The large capacity models are often prone to overfitting in a low-data regime and usually underperform [51]. For e.g. a ResNet-50 model [31] trained from scratch (via a softmax loss) for a 200-way fine-grained bird classification [86] using 30 examples-per-class overfits and yields 21.7% top-1 accuracy on a held-out validation set. In a single iteration of our approach, the same model gets 51.5% top-1 accuracy in a day. We take inspiration from prior art on growing networks [82, 87, 95]. These approaches slowly "grow" the network using unlabeled examples from similar distribution. In this work, we observe that we can quickly increase the capacity of model by streaming learning via a large amount of diverse unlabeled images. This is crucial specially when there is a possibility of a better representation but we could not explore them because of the lack of labeled and unlabeled data from similar distribution. It is also important to mention that because of the lack of labeled data for various tasks, many computer-vision approaches have been restricted to use the models designed for image classification specifically. Potentially, the use of domain agnostic unlabeled images in a streaming manner can enable us to even design better neural network architectures.

3. Method

Our streaming learning approach is a direct extension of semi-supervised learning algorithms. To derive our approach, assume we have access to an optimization routine that minimizes the loss on a supervised data set of labeled examples $(x, y) \in S$:

$$\text{Learn}(\mathcal{H}, S) \leftarrow \underset{F \in \mathcal{H}}{\operatorname{argmin}} \sum_{(x,y) \in S} \text{loss}(y, F(x)) \quad (1)$$

We will explore continually-evolving learning paradigms where the model class $\mathcal{H}$ grows in complexity over time (e.g., deeper models). We assume the gradient-based optimization routine is randomly initialized "from scratch" unless otherwise stated.

Semi-supervised learning: In practice, labeled samples are often limited. Semi-supervised learning assumes one has access to a large amount of unlabeled data $x \in U$. We specifically build on a family of deep semi-supervised approaches that psuedo-label unsupervised data U with a model trained on supervised data S [4, 35, 41]. Since these psuedo-labels will be noisy, it is common to pre-train on this large set, but fine-tune the final model on the pristine supervised set S [93]. Specifically, after learning an initial model F on the supervised set S:

1. Use F to pseudo-label U.

2. Learn a new model F' from random initialization on the pseudo-labelled U.

3. Fine-tune F' on S.

Iterative learning: The above 3 steps can be iterated for improved performance, visually shown in Fig. 3. It is natural to ask whether repeated iteration will potentially oscillate or necessarily converge to a stable model and set of pseudo-labels. The above iterative algorithm can be written as an approximate coordinate descent optimization [89] of a latent-variable objective function:

$$\min_{\{z\}, F \in \mathcal{H}} \sum_{(x,y) \in S} \text{loss}(y, F(x)) + \sum_{x \in U} \text{loss}(z, F(x)) \quad (2)$$

4

Algorithm 1: StreamLearning($S, \{U_t\}_{t=1}^T, \{\mathcal{H}_t\}_{t=1}^T$)

Input : S: Labeled dataset
$\{U_t\}_{t=1}^T$: T slices from unlabeled stream
$\{\mathcal{H}_t\}_{t=1}^T$: T hypothesis classes
Output: F
```
// Initialize the model on S
```
$F \leftarrow \text{Learn}(\mathcal{H}_1, S)$;
for $t \leftarrow 1$ *to* T **do**
```
    // Pseudo-label stream slice
```
  $U \leftarrow \{(x, F(x)) : x \in U_t\}$;
```
    // Pretrain model on U
```
  $F' \leftarrow \text{Learn}(\mathcal{H}_t, U)$;
```
    // Fine-tune model on S
```
  $F \leftarrow \text{Finetune}(F', S)$;
end

Step 1 optimizes for latent labels $\{z\}$ that minimize the loss, which are obtained by assigning them to the output of model $z := F(x)$ for each unlabeled example x. Step 2 and 3 optimize for F in a two-stage fashion. Under the (admittedly strong) assumption that this two-stage optimization finds the globally optimal F, the above will converge to a fixed point solution. In practice, we do not observe oscillations and find that model accuracy consistently improves.

Streaming learning: We point out two important extensions, motivated by the fact that the unsupervised set U can be massively large, or even an infinite stream (e.g., obtained by an online web crawler). In this case, Step 1 may take an exorbitant amount of time to finish labeling on U. Instead, it is convenient to "slice" up U into a streaming collection of unsupervised datasets U_t of manageable (but potentially growing) size, and simply replace U with U_t in Step 1 and 2. One significant benefit of this approach is that as U_t grows in size, we can explore larger and deeper models (since our approach allows us to pre-train on an arbitrarily large dataset U_t). In practice, we train a family of models $\mathcal{H}_t$ of increasing capacity on U_t. Our final streaming learning algorithm is formalized in Alg. 1.

4. Experiments

We first study the role of domain-agnostic unlabeled images in Section 4.1. We specifically study tasks where the data distribution of unlabeled images varies drastically from the labeled examples of the intended task. We then study the role of streaming learning in Section 4.2. We consider the well-studied task of fine-grained image classification here. We observe that one can dramatically improve the performance without using any task-specific knowledge. Finally, we study the importance of streaming learning from the perspective of a non-expert, i.e., cost in terms of time and money.

4.1. Role of Domain-Agnostic Unlabeled Images

We first contrast our approach with FixMatch [70] in Section 4.1.1. FixMatch is a recent state-of-the-art semi-supervised learning approach that use unlabeled images from similar distributions as that of the labeled data. We contrast FixMatch with SST in a setup where data distribution of unlabeled images differ from labeled examples. We then analyze the role of domain-agnostic unlabeled images to improve task-specific image classification in Section 4.1.2. The data distribution of unlabeled images dramatically differs from the labeled examples in this analysis. Finally, we extend our analysis to pixel-level tasks such as surface-normal estimation and semantic segmentation in Section 4.1.3. In these experiments, we use a million unlabeled images from ImageNet [64]. We use a simple softmax loss for image classification experiments throughout this work (unless otherwise stated).

4.1.1 Comparison with FixMatch [70]

We use two fine-grained image classification tasks for this study: (1) Flowers-102 [53] with 10 labeled examples per class; and (2) CUB-200 [86] with 30 labeled examples per class. The backbone model used is ResNet-18. We conduct analysis in Table 1 where we use the default hyperparameters from FixMatch [70] for analysis.

In specific, we use SGD optimizer with momentum 0.9 and the default augmentation for all experiments (except that FixMatch during training adopts both a strong and a weak (the default) version of image augmentation, whereas our approach only uses the default augmentation). For FixMatch, we train using lr 0.03, a cosine learning rate scheduling, L2 weight decay 5e-4, batch size 256 (with labeled to unlabeled ratio being 1:7) on 4 GPUs with a total of 80400 iterations. For our approach, we first train from scratch only on the labeled samples with the same set of hyperparameters as in FixMatch (with all 256 samples in the batch being labeled samples). From there we could already see that FixMatch sometimes does not match this naive training strategy. Then for our StreamLearning approach, we generate the pseudo-labels on the unlabeled set U_1 and trained for another 80400 iterations with lr 0.1 (decay to 0.01 at 67000 iteration), L2 weight decay 1e-4, batch size 256 on 4 GPUs. Finally, we finetuned on the labeled samples for another 80400 iterations with lr 0.1 (decay to 0.01 at 67000 iteration), L2 weight decay 1e-4, batch size 256 on 4 GPUs.

We also conduct analysis without hyperparameter tuning in Table 2, i.e., using default hyperparameters used in this work (see Appendix A.4). We observe FixMatch yields similar performance as the baseline model. Our approach, on the contrary, improves the performance over the baseline model even without specialized hyperparameters. Undoubtedly, spending expert human resources on hyperparameter tuning

helps us improve the performance. However, SST significantly outperforms FixMatch in both scenarios. Importantly, SST is task-agnostic and can be applied to pixel-level tasks as well without any modification.

Comparison with FixMatch

Task	Scratch	FixMatch [70]	U_1 (ours)
Flowers-102 [53]	58.21	53.00	**61.51**
CUB-200 [86]	44.24	51.24	**60.58**

Table 1. We contrast our approach with FixMatch [70] on two fine-grained image classification tasks. We use a million unlabeled images from ImageNet for this experiment. The backbone model used is ResNet-18. Our approach significantly outperforms FixMatch. We use the default hyperparameters from FixMatch [70].

Comparison with FixMatch (No Hyperparameter Tuning)

Task	Scratch	FixMatch [70]	U_1 (ours)
Flowers-102 [53]	45.49	43.19	**51.35**
CUB-200 [86]	44.03	44.93	**47.50**

Table 2. We contrast our approach with FixMatch [70] on two fine-grained image classification tasks. We use a million unlabeled images from ImageNet for this experiment. The backbone model used is ResNet-18. We abstain from hyperparameter tuning in these analysis and use the default hyperparameters used throughout this work (see Appendix A.4). We observe that FixMatch performs similar to the baseline model when the data distribution of unlabeled images is different from the labeled examples. Our approach, on the contrary, improves the performance over the baseline model.

4.1.2 Extreme-Task Differences

We use: (1) **EuroSat** [32] (satellite imagery) dataset for classifying satellite-captured images into distinct regions; (2) **ISIC2018** [16] (lesion diagnosis) for medical-image classification of skin diseases; and (3) **CropDiseases** [50] dataset which is a crop-disease classification task. We use 20 examples per class for each dataset and train the models from scratch. We provide details about the dataset and training procedure in the Appendix A.1.

Table 3 shows the performance for the three different tasks. We achieve significant improvement for each of them. We also show the performance of a pre-trained (using 1.2M labeled examples from ImageNet) model on these datasets. Guo et al. [28] suggested that fine-tuning a pre-trained model generally leads to best performances on these tasks. We observe that a simple random-gaussian initialization works as well despite trained using only a few labeled examples.

Crucially, we use unlabeled Internet images for learning a better representation on classification tasks containing

Task	pre-trained	init	U_1 (ours)
East-SAT [32]	68.93	70.57	73.58
Lesion [16]	45.43	44.86	50.86
Crop [50]	94.68	87.49	90.86

Table 3. **Extreme-Task Differences:** We analyse tasks that operate on specialized data distributions. We observe significant performance improvement despite using unlabeled streams of internet images. We also achieve performance competitive to the ImageNet-1k pre-trained model (again, trained with a large amount of labels). We use ResNet-18 for all experiments in the table.

classes that are extremely different to real-world object categories. Still, we see significant improvements.

4.1.3 Pixel Analysis

We extend our analysis to pixel-level prediction problems. We study surface-normal estimation using NYU-v2 depth dataset [66]. We intentionally chose this task because there is a large domain gap between NYU-v2 depth dataset and internet images of ImageNet-21k. We follow the setup of Bansal et al. [6, 7] for surface normal estimation because: (1) they demonstrate training a reasonable model from scratch; and (2) use the learned representation for downstream tasks. This allows us to do a proper comparison with an established baseline and study the robustness of the models. Finally, it allows us to verify if our approach holds for a different backbone-architecture (VGG-16 [68] in this case).

Evaluation: We use 654 images from the test set of NYU-v2 depth dataset for evaluation. Following [7], we compute six statistics over the angular error between the predicted normals and depth-based normals to evaluate the performance – **Mean**, **Median**, **RMSE**, **11.25°**, **22.5°**, and **30°** – The first three criteria capture the mean, median, and RMSE of angular error, where lower is better. The last three criteria capture the percentage of pixels within a given angular error, where higher is better.

Table 4 contrasts the performance of our approach with Bansal et al [6, 7]. They use a pre-trained ImageNet classification model for initialization. In this work, we initialize a model from random gaussian initialization (also known as scratch). The second-last row shows the performance when a model is trained from scratch. We improve this model using a million unlabeled images. The last row shows the performance after one iteration of our approach. We improve by 3-6% without any knowledge of surface normal estimation task. Importantly, we outperform the pre-trained ImageNet initialization. This suggests that we should not limit ourselves to pre-trained classification models that have access to large labeled datasets. We can design better neural network architectures for a task via SST.

Surface Normal Estimation (NYU Depthv2)						
Approach	Mean ↓	Median ↓	RMSE ↓	11.25° ↑	22.5° ↑	30° ↑
Bansal et al. [7]	19.8	12.0	28.2	47.9	70.0	77.8
Goyal et al. [27]	22.4	13.1	-	44.6	67.4	75.1
init (scratch)	21.2	13.4	29.6	44.2	66.6	75.1
U_1 (ours)	**18.7**	**10.8**	**27.2**	**51.3**	**71.9**	**79.3**

Table 4. We contrast the performance of our approach with Bansal et al [7, 6], which is the state-of-the-art given our setup. They use a pre-trained ImageNet classification model for initialization. In this work, we initialize a model from random gaussian initialization (also known as scratch). The third row shows the performance of a scratch-initialized model. We improve this model using one million unlabeled images. The last row shows the performance after one iteration of our approach. We improve by 3-6% without any domain-specific knowledge about the surface normal estimation task. Importantly, we outperform the pre-trained ImageNet initialization. We contrast our method with Goyal et al. [27] (second-row), which use $100M$ unlabeled images to train a generic representation via jigsaw puzzle [54] using a ResNet-50 model. Our model trained from scratch competes with their best performing model. This analysis suggests two things: (1) we can design better neural network architecture and does not have to limit ourselves to pre-trained classification models; and (2) SST can learn better models with two-orders less unlabeled data as compared to [27].

The details of the model and training procedure used in these experiments are available in the Appendix A.2. We have also provided analysis showing that we capture both local and global details without class-specific information.

Is it a robust representation? Bansal et al. [6] has used the model trained for surface-normal as an initialization for the task of semantic segmentation. We study if a better surface normal estimation means better initialization for semantic segmentation. We use the training images from PASCAL VOC-2012 [20] for semantic segmentation, and additional labels collected on 8498 images by [29] for this experiment. We evaluate the performance on the test set that required submission on PASCAL web server [1]. We report results using the standard metrics of region intersection over union (**IoU**) averaged over classes (higher is better). Refer to Appendix A.3 for details about training.

We show our findings in Table 5. We contrast the performance of surface-normal model trained from scratch (as in [6]) in the second row with our model in the third row. We observe a significant 2% performance improvement. This means better surface normal estimation amounts to a better initialization for semantic segmentation, and that we have a robust representation that can be used for down-stream tasks.

Can we improve semantic segmentation further? Can we still improve the performance of a task when we start from a better initialization other than scratch? We contrast the performance of the methods in the third row (init) to the fourth row (improvement in one-iteration). We observe

another significant 2.7% improvement in IoU. This conveys that we can indeed apply our insights even when starting from an initialization better than scratch. Finally, we observe that our approach has closed the gap between ImageNet (with class labels) pre-trained model and a self-supervised model to 3.6%.

4.2. Streaming Learning

We now demonstrate streaming learning for well studied fine-grained image classification in Section 4.2.1 where many years of research and domain knowledge (such as better loss functions [5, 8], pre-trained models, or hyperparameter tuning) has helped in improving the results. Here we show that streaming learning can reach close to that performance in few days without using any of this knowledge. In these experiments, we randomly sample from 14M images of ImageNet-21K [17] without ground truth labels as the unlabeled dataset.

4.2.1 Fine-Grained Image Classification

We first describe our experimental setup and then study this task using: (1) **Flowers-102** [53] that has 10 labeled examples per class; (2) **CUB-200** [86] that has 30 labeled examples per class; and (3) finally, we have also added analysis on a randomly sampled 20 examples per class from ImageNet-1k [64] (which we termed as **TwentyI-1000**). We use the original validation set [64] for this setup.

Model: We use the ResNet [31] model family as the hypothesis classes in Alg. 1, including ResNet-18, ResNet-34, ResNet-50, ResNext-50, and ResNext-101 [92]. The models are ranked in an increasing order of model complexity. Model weights are randomly generated by He initialization [30] (a random gaussian distribution) unless otherwise specified. We show in Appendix A.4 that training deeper neural networks with few labeled examples is non-trivial.

Learning F from the labeled sample S: Given the low-shot training set, we use the cross entropy loss to train the recognition model. We adopt the SGD optimizer with momentum 0.9 and a L2 weight decay of 0.0001. The initial learning rate is 0.1 for all experiments and other hyperparameters (including number of iterations and learning rate decay) can be found in Appendix A.4.

Learning F' from U with pseudo labels: Once we learn F, we use it to generate labels on a set of randomly sampled images from *ImageNet-21K* dataset to get pseudo-labelled U. Then we randomly initialize a new model F' as we do for F, then apply same network training for F' on U.

Finetuning F' on labeled sample S: After training F' on the pseudo-labeled U, we finetune F' on the original low-shot training set with the same training procedure and hyper-parameters. We use this finetuned model F' for test set evaluation.

Semantic Segmentation on VOC-2012																					
	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	bg	**IoU** ↑
scratch-init	62.3	26.8	41.4	34.9	44.8	72.2	59.5	56.0	16.2	49.9	45.0	49.7	53.3	63.6	65.4	26.5	46.9	37.6	57.0	40.4	85.2	**49.3**
normals-init	71.8	29.7	51.8	42.1	47.8	77.9	65.9	59.7	19.7	50.8	45.9	55.0	59.1	68.2	69.3	32.5	54.3	42.1	60.8	43.8	87.6	**54.1**
normalsStream-init	74.4	34.5	60.5	47.3	57.1	74.3	73.1	61.7	22.4	51.4	36.4	52.0	60.9	68.5	69.1	37.6	58.0	34.3	64.3	50.2	90.0	**56.1**
+one-iteration	82.2	35.1	62.0	47.4	62.1	76.6	74.1	62.7	23.9	49.9	47.0	55.5	58.0	74.9	73.9	40.1	56.4	43.6	65.4	52.8	90.9	**58.8**
pre-trained [6]	79.0	33.5	69.4	51.7	66.8	79.3	75.8	72.4	25.1	57.8	52.0	65.8	68.2	71.2	74.0	44.1	63.7	43.4	69.3	56.4	91.1	**62.4**

Table 5. The goal of this experiment is to study two things: **(1)** *Can task-specific representations learned on unlabeled streams generalize to other tasks?* This allows us to study the robustness of our learned representations. We consider the target task of semantic segmentation and the source task of surface-normal estimation. Segmentation networks initialized with surface-normal networks already outperform random initialization (row2 vs row1), and further improve by 2% when initialized with stream-trained networks (row3). **(2)** *Can we still further improve the performance of a task when starting from an initialization better than scratch?* We then perform one additional iteration of stream learning (row4 vs row3), resulting in another 2.7% improvement, closing the gap between ImageNet pre-training to 3.6%.

initialization

iteration-1

iteration-2

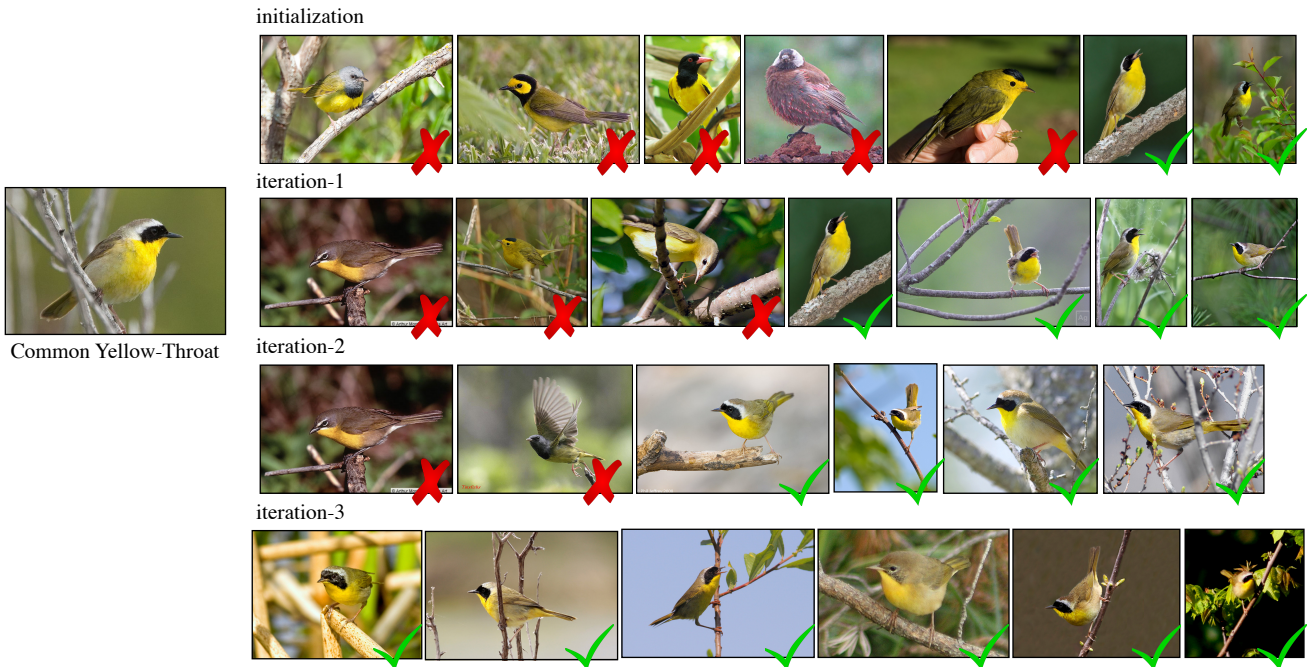iteration-3

Common Yellow-Throat

Figure 4. **Improvement in Recognizing Birds via Streaming Learning:** We qualitatively show improvement in recognizing a common yellow-throat (shown in left from CUB-200 dataset [86]). At **initialization**, the trained model confuses common yellow-throat with hooded oriole, hooded warbler, wilson rbler, yellow-breasted chat, and other similar looking birds. We get rid of false-positives with every **iteration**. At the the end of the third iteration, there are no more false-positives.

Streaming Schedule and Model Selection: We empirically observe that instead of training on entire unlabeled set U, we can slice up U into a streaming collections U_t for better performance. In these experiments, we use three iterations of our approach. We have 1M samples in U_1 (the same images as in *ImageNet-1K*), 3M samples in U_2, and 7M samples in U_3. We initialize the task using a ResNet-18 model (ResNet-18 gets competitive performance and requires less computational resources as shown in Table 11). We use a ResNext-50 model as F' to train on U_1 and U_2, and a ResNext-101 model to train on U_3. These design decisions are based on empirical and pragmatic observations shown in

Appendix A.5. Table 6 shows continuous improvement for various image-classification tasks at every iteration when using a few-labeled samples and training a model from scratch. We see similar trends for three different tasks. We are also able to bridge the gap between the popularly used pre-trained model (initialized using 1.2M labeled examples [64]) and a model trained from scratch without any extra domain knowledge or dataset/task-specific assumption.

4.2.2 Why Streaming Learning?

We study different questions here to understand our system. **What if we fix the model size in the iterations?** We ob-
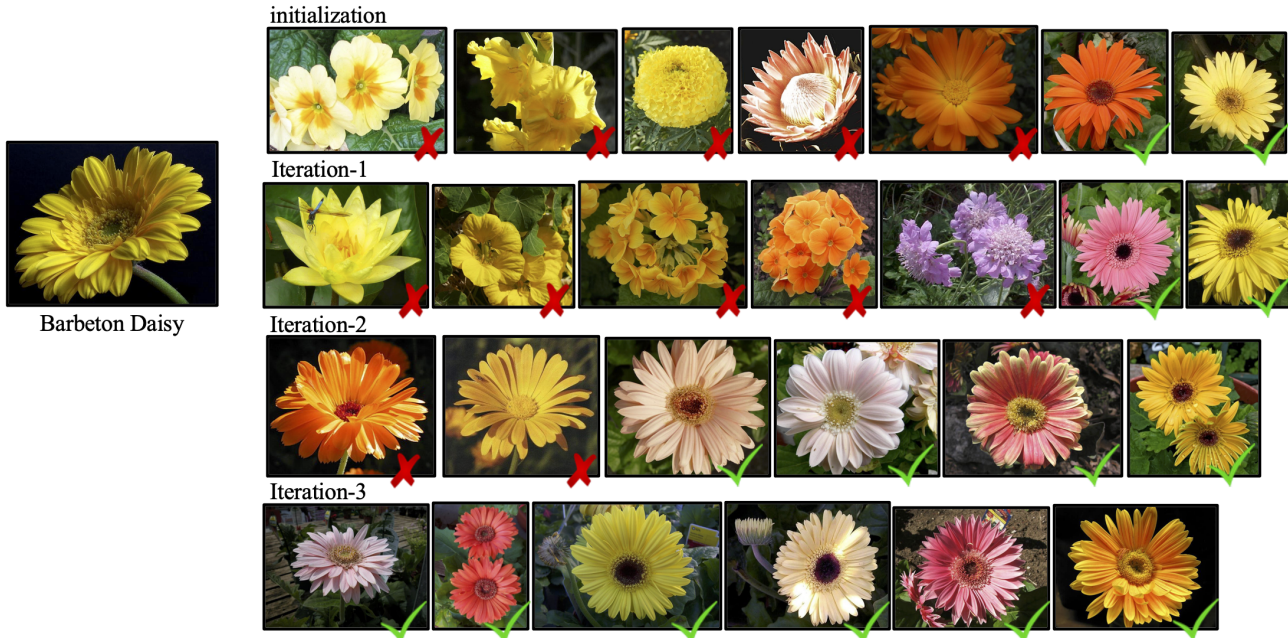
Figure 5. **Improvement in Recognizing Flowers via Streaming Learning:** We qualitatively show improvement in recognizing a barbeton daisy (shown in left from Flowers-102 dataset [53]). At **initialization**, the trained model confuses barbeton daisy with primula, water lily, daffodil, sweet william, and etc. With more iterations, the false positives become fewer.

Continuously Improving Image Classification						
Task	pre-trained	init	U_1	U_2	U_3	...
Flowers-102	89.12	45.49	54.19	65.25	72.79	...
CUB-200	75.29	44.03	53.73	57.11	66.10	...
TwentyI-1000	77.62	13.92	22.79	24.94	27.27	...

Table 6. We continuously improve the performance for Flowers-102, CUB-200, and TwentyI-1000, as shown by top-1 accuracy for each iteration. We achieve a large performance improvement for each iteration for all the tasks. This is due to the combination of both increasing unlabeled dataset and model size. Without any supervision, we can bridge the gap between an ImageNet-1k pretrained model and a model trained from scratch on Flowers-102 and CUB-200 dataset using a simple softmax loss.

serve that using deeper model could lead to faster improvement of the performance. For the *TwentyI-1000* experiment in section 4.2.1, we perform an ablative study by only training a ResNet-18 model, as shown in Table 7. We could still see the accuracy improving with more unlabeled data, but increasing model capacity turns out to be more effective.

What if we train without streaming? Intuitively, more iterations with our algorithm should lead to an increased performance. We verify this hypothesis by conducting another ablative study on *TwentyI-1000* experiment in section 4.2.1. In Table 8, we compare the result the result of training with three iterations (sequentially trained on U_1, U_2, U_3) with that

What if we use ResNet-18 for all experiments?					
Model	init	U_1	U_2	U_3	...
ResNet-18 only	13.92	19.61	21.22	22.13	...
StreamLearning	13.92	**22.79**	**24.94**	**27.27**	...

Table 7. We show that the top-1 validation accuracy on TwentyI-1000 for our **StreamLearning** approach (row 2) for each iteration, which increases the model capacity from ResNet-18 (init) to ResNext-50 (U_1 and U_2) to ResNext-101 (U_3). With **ResNet-18 only** (row 1), the performance gain is much slower.

of a single iteration (that concatenated all three slices together). Training on streams is more effective because improved performance on previous slices translates to more accurate pseudo-labels on future slices.

What if we train without streaming?					
Model	init	U_1	U_2	U_3	...
NoStreaming	13.92	**23.77**	–	–	–
StreamLearning	13.92	22.79	24.94	**27.27**	...

Table 8. We show that the top-1 validation accuracy on TwentyI-1000 for our **StreamLearning** approach (row 2) for each iteration, which increases the model capacity from ResNet-18 (init) to ResNext-50 (U_1 and U_2) to ResNext-101 (U_3). This result is compared to training with a **single iteration**, i.e, **NoStreaming**, that use ResNext-101 but with all the data.

9

Cost of Experiments: We now study the financial aspect of the streaming learning vs. single iteration via computing the cost in terms of time and money. We are given 11M unlabeled images and there are two scenarios: (1) train without streaming (U_1) using 11M images and ResNext-101; and (2) train in streams (U_1, U_2, U_3) of {1M, 3M, 7M} images using ResNext-50 for U_1 and U_2, and ResNext101 for U_3. For U_1, we train F' from scratch for 30 epochs. For U_2, we train F' from scratch for 20 epochs. For U_3, we train F' from scratch for 15 epochs. We could fit a batch of 256 images when using ResNext-50 on our 4 GPU machine. The average batch time is 0.39sec. Similarly, we could fit a batch of 128 images when using ResNext-101. The average batch time is 0.68sec. The total time for the first case (without streaming) is 486.96 hours (roughly 20 days). On the contrary, the total time for the streaming learning is 193.03 hours (roughly 8 days). Even if we get similar performance in two scenarios, we can get a working model in less than half time with streaming learning. A non-expert user can save roughly $1,470$ USD for a better performing model (60% reduction in cost), assuming they are charged 5 USD per hour of computation (on AWS).

5. Discussion

We present a simple and intuitive approach to semi-supervised learning on (potentially) infinite streams of un-labeled data. Our approach integrates insights from different bodies of work including self-training [19, 85], pseudo-labelling [41, 4, 35], continual/iterated learning [38, 39, 74, 75, 67], and few-shot learning [44, 28]. We demonstrate a number of surprising conclusions: (1) Unlabeled *domain-agnostic* internet streams can be used to significantly improve models for specialized tasks and data domains, including surface normal prediction, semantic segmentation, and few-shot fine-grained image classification spanning diverse domains including medical, satellite, and agricultural imagery. In this work, we use unlabeled images from ImageNet-21k [17]. While we do not use the labels, it is still a *curated* dataset that may potentially influence the performance. A crucial future work would be to analyze SST with truly in-the-wild image samples. This will also allow to go beyond the use of 14M images for learning better representation in a never ending fashion. (2) Continual learning on streams can be initialized with very *impoverished models* trained (from scratch) on tens of labeled examples. This is in contrast with much work in semi-supervised learning that requires a good model for initialization. (3) Contrary to popular approaches in semi-supervised learning that make use of massive compute resources for storing and processing data, streaming learning requires *modest computational infrastructure* since it naturally breaks up massive datasets into slices that are manageable for processing. From this perspective, continual learning on streams can help democratize research and development for scalable, lifelong ML.

A. Appendix

A.1. Extreme-Task Differences

Dataset: We randomly sample a 20-shot training set for each of the three datasets we present in the paper. For datasets without a test set, we curated a validation set by taking 10% of all samples from each category. Some of these datasets can be extremely different from natural images, and here we rank them in order of their similarity to natural images:

1. **CropDiseases** [50]. Natural images but specialized in agricultural industry. It has 38 categories representing diseases for different types of crops.

2. **EuroSat** [32]. Colored satellite images that are less similar to natural images as there is no perspective distortion. There is 10 categories representing the type of scenes, e.g., Forest, Highway, and etc.

3. **ISIC2018** [16]. Medical images for lesion recognition. There is no perspective distortion and no longer contains natural scenes. There are 7 classes representing different lesion. Because the dataset is highly unbalanced, we create a balanced test set by randomly sampling 50 images from each class.

Training details: We use ResNet-18 only for all experiments on the 3 cross-domain datasets, in order to isolate the effect of data. We also only do one iteration of our approach, but still we see substantial improvement. The unlabeled set U_1 is still the unlabeled version of *Imagenet-1K* dataset. We intentionally do this in order to contrast with the performance by finetuning an *ImageNet-pretrained* model with is pretrained using the same images but with additional $1.2M$ labels. We use SGD optimizer with momentum 0.9 and a L2 weight decay of 0.0001.

Learning F from the labeled sample S: For all these cross-domain few-shot datasets, we start with an initial learning rate of 0.1 while decaying it by a factor of 10 every 1500 epochs, and train for 4000 epochs.

Learning F' from U with pseudo labels: For U_1, we train F' from scratch for 30 epochs starting from learning rate 0.1, and decay it to 0.01 after 25 epochs.

Finetuning F' on the labeled sample S: We use the same training procedure when finetuning F' on S.

A.2. Surface Normal Estimation

Model and hyperparameters: We use the PixelNet model from [6] for surface normal estimation. This network architecture consists of a VGG-16 style architecture [68] and a multi-layer perceptron (MLP) on top of it for pixel-level prediction. There are 13 convolutional layers and three fully connected (*fc*) layers in VGG-16 architecture. The first two

Approach	Mean	Median	RMSE	$11.25°$	$22.5°$	$30°$
pre-trained [7]	19.8	12.0	28.2	47.9	70.0	77.8
init	21.2	13.4	29.6	44.2	66.6	75.1
init (until convergence)	20.4	12.6	28.7	46.3	68.2	76.4
U_1 (ours)	**18.7**	**10.8**	**27.2**	**51.3**	**71.9**	**79.3**

Table 9. **Can we improve scratch by training longer?** It is natural to ask if we can get better performance for training longer, crucially for a model trained from scratch. We observe that one can indeed get a slightly better performance by training for a long time. However, this improvement is negligible in comparisons to streaming learning.

fcs are transformed to convolutional filters following [47]. We denote these transformed *fc* layers of VGG-16 as conv-6 and conv-7. All the layers are denoted as $\{1_2, 1_2, 2_1, 2_2, 3_1, 3_2, 3_3, 4_1, 4_2, 4_3, 5_1, 5_2, 5_3, 6, 7\}$. We use hypercolumn features from conv-$\{1_2, 2_2, 3_3, 4_3, 5_3, 7\}$. An MLP is used over hypercolumn features with 3-fully connected layers of size $4,096$ followed by ReLU [40] activations, where the last layer outputs predictions for 3 outputs (n_x, n_y, n_z) with a euclidean loss for regression. Finally, we use batch normalization [34] with each convolutional layer when training from scratch for faster convergence. More details about the architecture/model can be obtained from [6].

Learning F from the labeled sample S: We use the above model, initialize it with a random gaussian distribution, and train it for NYU-v2 depth dataset [66]. The initial learning rate is set to 0.001, and it drops by a factor of 10 at step of $50,000$. The model is trained for $60,000$ iterations. We use all the parameters from [6], and have kept them fixed for our experiments to avoid any bias due to hyperparameter tuning.

Learning F' from U with pseudo labels: We use F trained above to psuedo-label 1M images, and use it to learn a F' initialized with random gaussian distribution and follows the same training procedure as F.

Finetuning F' on the labeled sample S: Finally, we finetune F' on S for surface normal estimation. The initial learning rate is set to 0.001, and it drops by a factor of 10 at step of $50,000$.

Can we improve *scratch* by training longer? It is natural to ask if we could improve the performance by training a model from scratch for more iterations. Table 9 shows the performance of training the scratch model for longer (until convergence). We observe that we do improve slightly over the model we use. However, this improvement is negligible in comparisons to streaming learning.

Can we capture both local and global information without class-specific information? One may suspect that a model initialized with the weights of pre-trained ImageNet classification model may capture more local information as the pre-training consists of class labels. Table 10 contrast the performance of two approaches on indoor scene furniture

Per-Object Surface Normal Estimation (NYU Depthv2)						
	Mean $\downarrow$	Median $\downarrow$	RMSE $\downarrow$	$11.25°$ $\uparrow$	$22.5°$ $\uparrow$	$30°$ $\uparrow$
chair						
Bansal et al. [7]	31.7	24.0	40.2	**21.4**	47.3	58.9
U_1 (ours)	**31.2**	**23.6**	**39.6**	21.0	**47.9**	**59.8**
sofa						
Bansal et al. [7]	20.6	15.7	26.7	35.5	66.8	78.2
U_1 (ours)	**20.0**	**15.2**	**26.1**	**37.5**	**67.5**	**79.4**
bed						
Bansal et al. [7]	19.3	13.1	26.6	44.0	70.2	80.0
U_1 (ours)	**18.4**	**12.3**	**25.5**	**46.5**	**72.7**	**81.7**

Table 10. We contrast the performance of our approach with the model fine-tuned using ImageNet (with class labels) on furniture categories, i.e. chair, sofa, and bed. Our approach outperforms prior art without any class information.

categories such as chair, sofa, and bed. The performance of our model exceeds prior art for local objects as well. This suggests that we can capture both local and global information quite well without class-specific information.

Finally, we qualitatively show improvement in estimating surface normal from a single 2D image in Figure 6.

A.3. Semantic Segmentation

We follow [6] for this experiment. The initial learning rate is set to 0.001, and it drops by a factor of 10 at step of $100,000$. The model is fine-tuned for $160,000$ iterations.

We follow the approach similar to surface normal estimation. We use the trained model on a million unlabeled images, and train a new model from scratch for segmentation. We used a batch-size of 5. The initial learning rate is also set to 0.001, and it drops by a factor of 10 at step of $250,000$. The model is trained for $300,000$ iterations. We then fine-tune this model using PASCAL dataset.

A.4. Fine-Grained Image Classiﬁ‹~cation

Datasets: We create few-shot versions of various popular image classification datasets for training. They are:

1. **Flowers-102** [53]. We train on the 10-shot versions of *Flowers* by randomly sampling 10 images per category from the training set. We report the top-1 accuracy on the test set for the 102 flower categories.

2. **CUB-200** [86]. We take 30 training examples per category from the Caltech UCSD Bird dataset and report the top-1 accuracy on the test set for the 200 birds categories.

3. **TwentyI-1000** [64] (ILSVRC 2012 Challenge) with 1000 classes. Specially, we train on a 20-shot version of *ImageNet-1K*. We report the top-1 validation set accuracy on the 1000 classes as commonly done in literature [91].
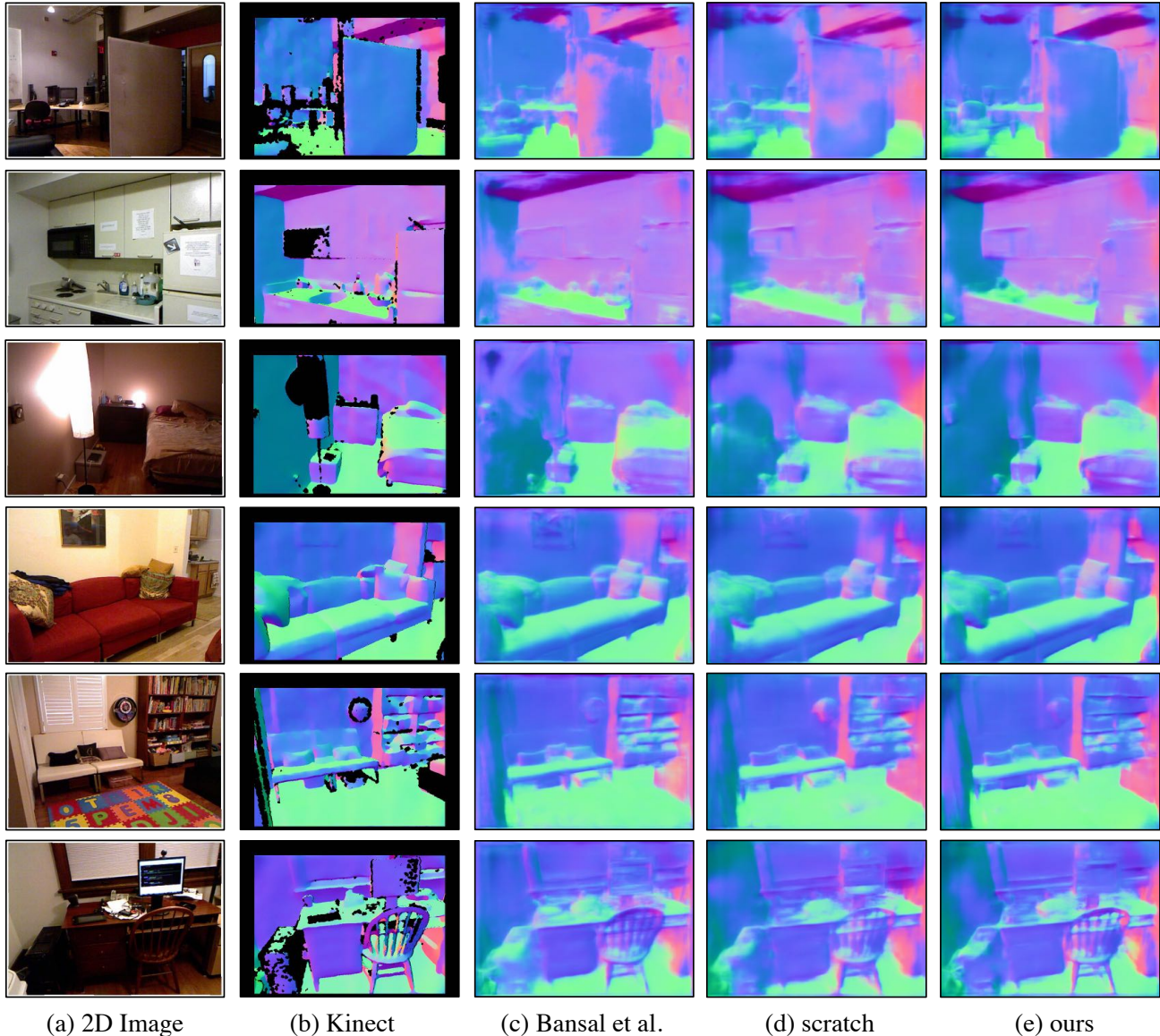
| (a) 2D Image | (b) Kinect | (c) Bansal et al. | (d) scratch | (e) ours |

Figure 6. **Surface Normal Estimation:** For a given single 2D image (shown in (**a**)), we contrast the performance of various models. Shown in (**c**) are the results from prior work [7, 6] using a model pretrained with *ImageNet-1K* labels; (**d**) shows a model trained from scratch starting from random gaussian initialization; and finally (**e**) shows the result of our StreamLearning approach. The influence of unlabeled data can be gauged by improvements from (**d**) to (**e**). By utilizing diverse unlabeled data, we can get better performance without any additional supervision. For reference, we also show ground truth normals from kinect in (**b**).

Model and hyperparameters: We experiment with the ResNet [31] model family, including ResNet-18, ResNet-34, ResNet-50, ResNext-50, and ResNext-101 [92]. The models are ranked in an increasing order of model complexity. The initial model weights are randomly generated by He initialization [30], which is the PyTorch default initialization scheme. For all image classification experiments, we adopt the SGD optimizer with momentum 0.9 and a L2 weight decay of 0.0001. We use an initial learning rate of 0.1 for both finetuning on S and training on U.

Learning F from the labeled sample S: For **Flowers-102** (10-shot), we decay the learning rate by a factor of 10 every 100 epochs, and train for a total of 250 epochs. For **CUB-200** (30-shot), we decay the learning rate by a factor of 10 every 30 epochs, and train for 90 epochs. For **TwentyI-1000**, we decay the learning rate by a factor of 10 every 60 epochs, and train for a total of 150 epochs.

Streaming Schedule: We simulate an infinite unlabeled stream U by randomly sampling images from *ImageNet-21K*. In practice, we slice the data into a streaming collections

One-stage models trained from scratch			
Model	Flowers-102	CUB-200	TwentyI-1000
Resnet-18	**45.49**	**44.03**	**13.92**
Resnet-34	42.64	**44.17**	**14.23**
Resnet-50	20.82	21.73	12.93
Resnext-50	31.34	28.37	11.87
Resnext-101	34.18	32.31	13.35

Table 11. We show performance of various models when trained from scratch. It is non-trivial to train a deep neural network with a few labeled examples as shown in this analysis. Despite increasing the capacity of the models and training them for longer, we do not observe any performance improvement.

U_t. We have 1M samples in U_1, 3M samples in U_2, and 7M samples in U_3. We intentionally make U_1 the unlabeled version of *Imagenet-1K* dataset for comparison with other works that use the labeled version of *Imagenet-1K*.

Model Selection: We initialize the task using a ResNet-18 model because it achieved great generalization performance when training from scratch compared to deeper models and only costs modest number of parameters. We use a ResNext-50 model as F' to train on U_1 and U_2, and a ResNext-101 model to train on U_3. These design decisions are based on empirical and pragmatic observations we provided in Appendix A.5.

Learning F' from U with pseudo labels: For U_1, we train F' from scratch for 30 epochs starting from learning rate 0.1, and decay it to 0.01 after 25 epochs. For U_2, we train F' from scratch for 20 epochs and decay the learning rate to 0.01 after 15 epochs. For U_3, we train F' from scratch for 15 epochs and decay the learning rate to 0.01 after 10 epochs.

Finetuning F' on the labeled sample S: We use the same training procedure when finetuning F' on S.

A.5. Ablative Analysis

We study different questions here to understand the working of our system.

What is the performance of models trained from scratch? We show performance of various models when trained from scratch in Table 11. We observe that training deeper neural networks from random initialization with few labeled examples is indeed non-trivial. Therefore, our approach helps deeper networks generalize better in such few shot settings.

Why do we use ResNext-50 for U_1 and U_2? We show in Table 12 that ResNext-50 outperforms ResNet-18 in first iteration to justify the model decision of our stream learning approach. Note that this is not saying ResNext-50 is the best performing model among all possible choices. For instance, ResNext-101 slightly outperforms ResNext-50 (around 1%

improvement) on the first two iterations, but we still use ResNext-50 for U_1 and U_2 for pragmatic reasons (faster to train and save more memory). In practice, one can trade off generalization performance and training speed by select the most suitable model size just like what we did in this paper.

Performance after U_1: ResNet-18 or ResNext-50?			
Model	CUB-200	Flowers-102	TwentyI-1000
ResNet-18	51.35	47.50	19.61
ResNext-50	**53.73**	**54.19**	**22.79**

Table 12. We show that the top-1 validation accuracy on all fine-grained classification datasets with our approach after the first iteration (U_1 with 1M unlabeled images) training with ResNet-18 or ResNext-50. We can see that ResNext-50 consistently outperforms ResNet-18 across all tasks.

References

[1] Pascal voc server. https://host.robots.ox.ac.uk:8080//. 7

[2] Woo-Kyoung Ahn and William F Brewer. Psychological studies of explanationbased learning. In *Investigating explanation-based learning*. Springer, 1993. 3

[3] Woo-Kyoung Ahn, Raymond J Mooney, William F Brewer, and Gerald F DeJong. Schema acquisition from one example: Psychological evidence for explanation-based learning. Technical report, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1987. 3

[4] Eric Arazo, Diego Ortego, Paul Albert, Noel E OConnor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *IEEE IJCNN*, 2020. 2, 3, 4, 10

[5] Idan Azuri and Daphna Weinshall. Learning from small data through sampling an implicit conditional generative latent optimization model. In *IEEE ICPR*, 2020. 2, 3, 7

[6] Aayush Bansal, Xinlei Chen, Bryan Russell, Abhinav Gupta, and Deva Ramanan. PixelNet: Representation of the pixels, by the pixels, and for the pixels. *arXiv:1702.06506*, 2017. 6, 7, 8, 10, 11, 12

[7] Aayush Bansal, Bryan Russell, and Abhinav Gupta. Marr Revisited: 2D-3D model alignment via surface normal prediction. In *CVPR*, 2016. 6, 7, 11, 12

[8] Bjorn Barz and Joachim Denzler. Deep learning on small datasets without pre-training using cosine loss. In *IEEE/CVF WACV*, 2020. 2, 3, 7

[9] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *ICLR*, 2020. 2, 3

[10] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 2, 3

[11] Shengcao Cao, Xiaofang Wang, and Kris M. Kitani. Learnable embedding space for efficient neural architecture compression. In *ICLR*, 2019. 3

[12] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, 2018. 3

[13] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. *IEEE Trans. NNLS*, 2009. 3

[14] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020. 2, 4

[15] Yanbei Chen, Xiatian Zhu, and Shaogang Gong. Semi-supervised deep learning with memory. In *ECCV*, 2018. 2, 3

[16] 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:1902.03368*, 2019. 4, 6, 10

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 7, 10

[18] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 3

[19] Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Ves Stoyanov, and Alexis Conneau. Self-training improves pre-training for natural language understanding. *arXiv:2010.02194*, 2020. 2, 10

[20] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 2010. 3, 7

[21] Zeyu Feng, Chang Xu, and Dacheng Tao. Self-supervised representation learning from multi-domain data. In *CVPR*, 2019. 2, 3

[22] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 2, 3, 4

[23] Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. In *ICML*, 2019. 3

[24] John H Flavell. *Cognitive development.* prentice-hall, 1977. 2

[25] David F. Fouhey, Abhinav Gupta, and Martial Hebert. Data-driven 3D primitives for single image understanding. In *ICCV*, 2013. 4

[26] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *CoRR*, 2018. 3

[27] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *ICCV*, 2019. 7

[28] Yunhui Guo, Noel CF Codella, Leonid Karlinsky, John R Smith, Tajana Rosing, and Rogerio Feris. A new benchmark for evaluation of cross-domain few-shot learning. In *ECCV*, 2020. 2, 4, 6, 10

[29] B. Hariharan, P. Arbel'ez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, 2011. 7

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 7, 12

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4, 7, 12

[32] 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*, 2019. 4, 6, 10

[33] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 3

[34] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 11

[35] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *CVPR*, 2019. 2, 3, 4, 10

[36] Hamid Izadinia, Bryan C. Russell, Ali Farhadi, Matthew D. Hoffman, and Aaron Hertzmann. Deep classifiers from image tags in the wild. In *Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions*. ACM, 2015. 3

[37] Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *ECCB*. Springer, 2016. 3

[38] Simon Kirby. Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation*, 2001. 3, 10

[39] Simon Kirby, Tom Griffiths, and Kenny Smith. Iterated learning and the evolution of language. *Current opinion in neurobiology*, 2014. 3, 10

[40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 3, 11

[41] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013. 2, 3, 4, 10

[42] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 2, 4

[43] Boaz Lerner, Guy Shiran, and Daphna Weinshall. Boosting the performance of semi-supervised learning with unsupervised clustering. *arXiv preprint arXiv:2012.00504*, 2020. 2, 3

[44] Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. Learning to self-train for semi-supervised few-shot classification. In *NeurIPS*, 2019. 3, 10

[45] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 2017. 3

14

[46] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 3

[47] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional models for semantic segmentation. In *CVPR*, 2015. 11

[48] Yuchen Lu, Soumye Singhal, Florian Strub, Olivier Pietquin, and Aaron Courville. Countering language drift with seeded iterated learning. *arXiv:2003.12694*, 2020. 3

[49] Yuchen Lu, Soumye Singhal, Florian Strub, Olivier Pietquin, and Aaron Courville. Supervised seeded iterated learning for interactive language learning. In *Proc. of EMNLP*, 2020. 3

[50] Sharada P Mohanty, David P Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016. 6, 10

[51] Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *CVPR*, 2020. 2, 3, 4

[52] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000. 3

[53] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008. 1, 5, 6, 7, 9, 11

[54] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016. 7

[55] Cheng Perng Phoo and Bharath Hariharan. Self-training for few-shot transfer across extreme task differences. In *ICLR*, 2021. 2, 3

[56] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *CVPR*, 2018. 3

[57] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. 3

[58] Ilija Radosavovic, Piotr Dollár, Ross Girshick, Georgia Gkioxari, and Kaiming He. Data distillation: Towards omni-supervised learning. In *CVPR*, 2018. 2, 3

[59] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, 2007. 3

[60] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *NeurIPS*, 2019. 3

[61] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 3

[62] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 3

[63] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 3

[64] 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. *IJCV*, 2015. 3, 4, 5, 7, 8, 11

[65] H Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Trans. IT*, 1965. 2

[66] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 3, 4, 6, 11

[67] Daniel L Silver, Qiang Yang, and Lianghao Li. Lifelong machine learning systems: Beyond learning algorithms. In *AAAI spring symposium series*, 2013. 3, 10

[68] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3, 6, 10

[69] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 2, 3, 4

[70] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. 2, 3, 5, 6

[71] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017. 3

[72] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 2, 3, 4

[73] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICLR*, 2019. 3

[74] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In *NeurIPS*, 1996. 3, 10

[75] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998. 3, 10

[76] 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*, 2018. 4

[77] Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. In *ICLR*, 2020. 2, 4

[78] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine Learning*, 2020. 2

[79] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 2, 4

[80] Bram Wallace and Bharath Hariharan. Extending and analyzing self-supervised learning across domains. In *ECCV*, 2020. 2, 3

[81] Matthew Wallingford, Aditya Kusupati, Keivan Alizadeh-Vahid, Aaron Walsman, Aniruddha Kembhavi, and Ali Farhadi. In the wild: From ml models to pragmatic ml systems. *arXiv:2007.02519*, 2020. 3

[82] Guangcong Wang, Xiaohua Xie, Jianhuang Lai, and Jiaxuan Zhuo. Deep growing learning. In *ICCV*, 2017. 4

[83] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *CVPR*, 2015. 3, 4

[84] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, 2018. 3

[85] Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. Theoretical analysis of self-training with deep networks on unlabeled data. *arXiv:2010.03622*, 2020. 2, 10

[86] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 4, 5, 6, 7, 8, 11

[87] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *NeurIPS*, 2016. 4

[88] Davis Wertheimer and Bharath Hariharan. Few-shot learning with localization in realistic settings. In *CVPR*, 2019. 3

[89] Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015. 4

[90] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. In *NeurIPS*, 2020. 2, 3

[91] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020. 2, 3, 11

[92] Saining Xie, Ross Girshick, Piotr Dollr, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 7, 12

[93] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv:1905.00546*, 2019. 2, 3, 4

[94] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Association for Computational Linguistics*, 1995. 2

[95] Qifei Zhang and Xiaomo Yu. Growingnet: An end-to-end growing network for semi-supervised learning. *Computer Communications*, 151:208–215, 2020. 4

[96] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. *ECCV*, 2016. 3

[97] Yuting Zhang, Kibok Lee, and Honglak Lee. Augmenting supervised neural networks with unsupervised objectives for large-scale image classification. In *ICML*, 2016. 3

[98] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE TPAMI*, 2017. 3

[99] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 2018. 3

[100] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005. 3

[101] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018. 3