Journal of Machine Learning Research 24 (2023) 1-77 Submitted 11/22; Revised 8/23; Published 8/23

Nevis'22: A Stream of 100 Tasks Sampled from 30 Years of Computer Vision Research

Jörg Bornschein* BORNSCHEIN@DEEPMIND.COM
Alexandre Galashov* AGALASHOV@DEEPMIND.COM
Ross Hemsley* RHEMSLEY@DEEPMIND.COM
Amal Rannen-Triki* ARANNEN@DEEPMIND.COM
Yutian Chen YUTIANC@DEEPMIND.COM
Arslan Chaudhry ARSLANCH@DEEPMIND.COM
Xu Owen He HEXU@DEEPMIND.COM
Arthur Douillard DOUILLARD@DEEPMIND.COM
Massimo Caccia† MASSIMO.P.CACCIA@GMAIL.COM
Qixuan Feng QIXUAN@DEEPMIND.COM
Jiajun Shen JIAJUNS@DEEPMIND.COM
Sylvestre-Alvise Rebuffi SYLVESTRE@DEEPMIND.COM
Kitty Stacpoole KSTACPOOLE@DEEPMIND.COM
Diego de las Casas DIEGOLASCASAS@DEEPMIND.COM
Will Hawkins WILLHAWKINS@DEEPMIND.COM
Angeliki Lazaridou ANGELIKI@DEEPMIND.COM
Yee Whye Teh YWTEH@DEEPMIND.COM
Andrei A. Rusu ANDREI@DEEPMIND.COM
Razvan Pascanu RAZP@DEEPMIND.COM
Marc'Aurelio Ranzato RANZATO@DEEPMIND.COM
DeepMind ‡

Editor: Christoph Lampert

Abstract

A shared goal of several machine learning communities like continual learning, meta-learning and transfer learning, is to design algorithms and models that efficiently and robustly adapt to unseen tasks. An even more ambitious goal is to build models that never stop adapting, and that become increasingly more efficient through time by suitably transferring the accrued knowledge. Beyond the study of the actual learning algorithm and model architecture, there are several hurdles towards our quest to build such models, such as the choice of learning protocol, metric of success and data needed to validate research hypotheses. In this work, we introduce the **N**ever-**E**nding **VI**sual-classification **S**tream (Nevis'22), a benchmark consisting of a stream of over 100 visual classification tasks, sorted chronologically and extracted from papers sampled uniformly from computer vision

∗. Equal contribution
†. Current affiliation: Mila - Quebec AI Institute. Work done while interning at DeepMind.
‡. For questions about the benchmark, please email us at nevis@deepmind.com or write to us at 14-18 Handyside Street, King's Cross, London, N1C 4DN. More details are provided in Appendix A.

©2023 Jörg Bornschein, Alexandre Galashov, Ross Hemsley, Amal Rannen-Triki, Yutian Chen, Arslan Chaudhry, Xu Owen He, Arthur Douillard, Massimo Caccia, Qixuan Feng, Jiajun Shen, Sylvestre-Alvise Rebuffi, Kitty Stacpoole, Diego de las Casas, Will Hawkins, Angeliki Lazaridou, Yee Whye Teh, Andrei A. Rusu, Razvan Pascanu, Marc'Aurelio Ranzato.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v24/22-1345.html.

proceedings spanning the last three decades. The resulting stream reflects what the research community thought was meaningful at any point in time, and it serves as an ideal test bed to assess how well models can adapt to new tasks, and do so better and more efficiently as time goes by. Despite being limited to classification, the resulting stream has a rich diversity of tasks from OCR, to texture analysis, scene recognition, and so forth. The diversity is also reflected in the wide range of dataset sizes, spanning over four orders of magnitude. Overall, Nevis'22 poses an unprecedented challenge for current sequential learning approaches due to the scale and diversity of tasks, yet with a low entry barrier as it is limited to a single modality and well understood supervised learning problems. Moreover, we provide a reference implementation including strong baselines and an evaluation protocol to compare methods in terms of their trade-off between accuracy and compute. We hope that Nevis'22 can be useful to researchers working on continual learning, meta-learning, AutoML and more generally sequential learning, and help these communities join forces towards more robust models that efficiently adapt to a never ending stream of data[1].

Keywords: benchmark, transfer learning, continual learning, meta-learning, AutoML.

1. Introduction

The machine learning community has focused extensively on the *stationary* batch setting, for which there exists a static and unchanging data distribution used to sample fixed training and test sets from (Vapnik, 1998). This has enabled the rigorous evaluation of learning systems, and driven unprecedented progress over the last four decades, across a wide range of domains (e.g. LeCun et al., 2015; Jumper et al., 2021; Brown et al., 2020; Alayrac et al., 2022). Throughout this journey, researchers have spent a considerable amount of time and compute developing algorithmic and architectural improvements, adapting methods to new application domains, and developing insights into how to transfer their knowledge and know-how to new and more challenging settings.

Over the past decade, there has been a surge of interest in the design of learning algorithms that generalize not only to novel examples, but also to entirely new tasks (Zhai et al., 2019; Wald et al., 2021; Gulrajani and Lopez-Paz, 2020; Triantafillou et al., 2020). This line of research relates to efforts to automate the design process of architectures (Bai et al., 2021; Ardywibowo et al., 2020) and improve learning algorithms (Arjovsky et al., 2019; Triantafillou et al., 2021). Broadly speaking, the goal of this new endeavors is to understand the principles and to design learning algorithms that enable further adaptation after training. In fact, training never really ends. The system observes a never-ending sequence of tasks and the question is how it can adapt faster and better over time. Can it succeed by leveraging its ever increasing knowledge of the world acquired through its past experiences, while limiting as much as possible human intervention?

There are several open questions in this research area, from how to represent knowledge, to how to accrue knowledge over time, how to retain computational efficiency, etc. In this work we focus on the methodology and data used as playground for advancing research in this area. First, we consider a stream of vision classification tasks. Each such task is very well understood, the only remaining challenge is how to automatically and efficiently learn such tasks in sequence. Second, we take a *hindsight* approach to the benchmark construction process. Our objective is to eventually deploy a system that is capable of

1. Implementations have been made available at `https://github.com/deepmind/dm_nevis`.

automatically learning whatever task the research community comes up with and, by doing so, to become more apt at solving any other future task. We therefore build a stream by sorting chronologically the tasks that the research community has introduced and used over the last three decades. We then assess whether models that have learned on all the tasks up to time t, can better learn the next task, and whether learning becomes more effective and efficient for larger values of t.

This construction process stands in stark contrast to how current benchmarks are built. These are often very small scale which prevents the assessment of efficiency of learning, they are very homogeneous which prevents the assessment of robustness of learning, and they are built out of a small number of hand picked tasks which might poorly represent the task distribution of interest to our community.

This motivates us to introduce Nevis'22, a challenging stream which comprises 106 tasks, all representing publicly available datasets from the last 30 years of computer vision research. By construction, Nevis'22 tracks what the vision community has deemed interesting over time, since tasks are sorted according to the year in which they appeared in publications. Over time, new and more challenging domains are considered, datasets get larger, and overall there are more opportunities to transfer knowledge from an ever growing set of related tasks.

As an indirect measure of whether a system is capable of accruing knowledge over time, we assess performance not just in terms of final error rate, but also compute needed to reach such performance. The assumption is that, if a method can transfer knowledge from related past tasks, then it can quickly learn the next task using less compute.

We believe Nevis'22 should appeal to and challenge researchers in several communities. It should attract researchers in continual learning (Ring, 1994; Thrun, 1994) because the stream is non-stationary. Some of the tasks are repeated over time, providing a natural opportunity to measure forgetting and forward transfer (Lopez-Paz and Ranzato, 2017; Chaudhry et al., 2018; Schwarz et al., 2018; Hadsell et al., 2020; Parisi et al., 2019). It should empower researchers in meta-learning (Thrun and Pratt, 1998) because there is a rich structure across tasks, which should enable the study of learning-to-learn. Finally, it should be useful to researchers in AutoML (Thornton et al., 2013) as each task has to be solved in a black box manner, without humans in the loop. Since our metrics include the compute used during hyper-parameter search, Nevis'22 incentivizes the development of efficient approaches for algorithm, architecture and hyper-parameter search. For the very same reasons, however, Nevis'22 also constitutes a challenge, as it requires tools from each of these communities. Moreover, Nevis'22 is the first benchmark simulating supervised never-ending learning at this scale, and with such rich and diverse set of realistic tasks. Nevis'22 is accompanied by code to reproduce the stream, the training and evaluation protocols, and representative baselines we have considered. We summarize the main findings in Table 1 with more detailed discussion in Section 5.6 and in the Appendix.

2. Related Work

In this section we put our work in the broader context of the literature with a twofold goal. First, we relate the Nevis'22 learning setting with existing learning frameworks such as continual learning, meta-learning and AutoML. Second, we contrast Nevis'22 with existing benchmarks and highlight its unique features. Tables 2 and 3 provide a high level overview.

Findings	**References**
Nevis'22 enables the comparison of methods in terms of their compute-performance trade-off	Fig. 4
Nevis'22 favors methods that leverage knowledge transfer across tasks (e.g., various forms of fine-tuning)	Section 5.6, Figure 4, Figure 5, Section E.1, Figure 18, Figure 16, Figure 6
Nevis'22 enables the study of how to use and adapt pretrained representations	Section 5.6, Figure 4, Figure 5, Section E.2
Nevis'22 shows that current methods are not capable of transferring from a large number of smaller datasets	Tab.5
Nevis'22 supports fine-grained analysis (domain, forward-transfer, etc.)	Fig. 6, 11

Table 1: Summary of main findings. For more information, please refer to Section 5.6. Additional results are given in Appendix.

	sequential	causal	memory restrictions	compute in the metric
continual learning	yes	no	yes	no
meta-learning	no	-	no	no
auto-ml	no	-	no	no
lifelong auto-ml	yes	no	no	yes
Nevis'22	**yes**	**yes**	**no**	**yes**

Table 2: Comparing learning frameworks across several axes, namely whether the learner observes tasks in sequence, it has access to future task while doing task specific hyper-parameter search (i.e., the model is allowed to do several passes over the stream), it has memory restrictions when accessing data and model parameters of past tasks, and whether compute is accounted in the evaluation. Note that this is an oversimplification, as often papers use intermediate setups.

Continual (or Lifelong) Learning studies the question of learning under a non-stationary data distribution (Silver et al., 2013; Chen and Liu, 2018; Hadsell et al., 2020; Parisi et al., 2019). It typically assumes a series of tasks. The objective is to learn sequentially while achieving a list of desiderata ranging from avoiding catastrophic forgetting (McCloskey and Cohen, 1989), to leveraging forward or backward transfer[2] (Lopez-Paz and Ranzato, 2017). Additional restrictions are typically considered, such as preventing access to previous data, limiting or accounting for the use of compute or memory. Given the multitude of potential desiderata and trade-offs of interest (Hadsell et al., 2020), the literature has flourished with

2. A model has positive backward (forward) transfer when performance on a past (future) task improves upon learning a task.

	sequential	large-scale	diversity	compute in metric
MNIST (LeCun et al., 1998b) variants	yes	no	no	no
CIFAR (CI41) variants	yes	no	no	no
CTrL (Veniat et al., 2021)	yes	yes	no	yes
CLOC (Cai et al., 2021)	yes	yes	no	no
CLEAR (Lin et al., 2021)	yes	yes	no	no
VTAB (Zhai et al., 2019)	no	yes	yes	no
Meta-Dataset(Triantafillou et al., 2020)	no	yes	yes	no
Nevis'22	**yes**	**yes**	**yes**	**yes**

Table 3: Comparing benchmarks made of several classification tasks.

a considerable number of specialized benchmarks, each targeting a different scenario. In this work, we make the following central assumption: The learner cannot access data from novel future tasks. However, accessing past data (or even past models) is permitted, although the compute cost of doing so will be taken into account in the final reporting. Our rationale is that, in modern applications of machine learning, memory for storing training data is cheap relative to compute and time. We therefore focus on leveraging forward transfer for future tasks of interest to the community, rather than avoiding catastrophic forgetting or imposing data storage limitations.

In continual Reinforcement Learning (RL) (Ring, 1994; Khetarpal et al., 2020), the non-stationarity is either imposed by changing environments, or it emerges from the interaction with the environment. While RL provides a natural test-bed for continual learning, it also makes it difficult to separate the challenges raised by exploration and learning with sparse rewards from the core continual learning problem of accruing knowledge over time. Attempts to studying the question of forward transfer have nonetheless been made (Wolczyk et al., 2021). Similarly in the language domain, there have been studies on continuous training of language models with related benchmarks defined on sorted streams of text (Liska et al., 2022; Jang et al., 2022). Although very interesting, the lack of clear task structure or distinctions makes it difficult to assess when new concepts are introduced in data streams, and when the system is expected to learn new capabilities or skills.

In vision research, most existing benchmarks focus on measuring catastrophic forgetting and are derived from popular datasets such as MNIST (LeCun et al., 1998b), CIFAR-10 (Krizhevsky, 2009b), ImageNet (Deng et al., 2009b) or Omniglot (Lake et al., 2015), whereby a stream is created by partitioning the data into disjoint subsets. This construction however greatly limits the diversity of the resulting stream. There are however exceptions. The Core50 dataset (Lomonaco and Maltoni, 2017) specifically collected realistic images of objects under different poses, to test continual learning capabilities in a setting most relevant for robotics. The CLEAR benchmark (Lin et al., 2021) looks at temporal evolution of a set of visual concepts. CLOC (Cai et al., 2021) is a geo-localization task with a large collection of chronologically ordered images. Once again, these benchmarks target the setting of a single non-stationary task, as opposed to a sequence of a diverse set of tasks. Forward transfer has become a more prominent goal of CL through benchmarks such as CTrL (Veniat et al.,

2021), which is however limited in its scale and diversity, because it is entirely derived from just a handful of small datasets.

Meta-learning assumes access to a distribution of tasks (Thrun and Pratt, 2012; Finn, 2018; Hospedales et al., 2022). The goal is to learn from the observed tasks a mechanism that allows efficient learning on hold-out tasks from the same distribution. Most popular benchmarks like VTAB (Zhai et al., 2019) and Meta-Dataset (Triantafillou et al., 2020) focus on few-shot learning, while Nevis'22 considers a variety of dataset sizes (including some with a handful of examples) and goes beyond a few steps of adaptation to characterize performance efficiency trade-offs. For example, Nevis'22 accounts for compute in addition to error rate.

Much of the field of AutoML is concerned with the *automatic* discovery of algorithms, architectures and optimisers for a given new task. Auto-ML is mostly focused on tabular tasks and shallow predictors. Benchmarks are often derived from the OpenML platform (Vanschoren et al., 2013). The major limitation of current AutoML approaches is their computational cost, since one evaluation requires a full training run with a particular hyper-parameter setting. For instance, naïve neural architecture search would be too costly if applied within the inner loop of Nevis'22, as there are over 100 tasks. More recently, there has been excitement around lifelong AutoML (Feurer et al., 2015; Lindauer and Hutter, 2018), but again, evaluation has been limited to very few and very similar tasks. Nevis'22 gives a more natural playground to explore ways to transfer knowledge at the level of the meta-learner thanks to shared structure across tasks. Moreover, Nevis'22 sets incentives towards the development of more efficient AutoML methods because the evaluation accounts for compute spent during hyper-parameter search and prizes more parsimonious meta-learners.

Transfer learning (Pan and Yang, 2010; Bengio, 2012; Weiss et al., 2016; Tan et al., 2018; Zhuang et al., 2020) is a general and well studied paradigm for addressing the problem of leveraging one or a few related source tasks to improve the learning of a known target task. Techniques such as self-supervised learning (Chen et al., 2020; Grill et al., 2020) and large-scale pretraining (Jia et al., 2021; Radford et al., 2021) are recent examples of successful approaches in computer vision, see (Jaiswal et al., 2020) for a survey. The sequential aspect of data acquisition is often neglected in the transfer learning setting. Nevis'22 provides a good test bed for testing transfer learning ideas at scale, and in a more realistic setting, where the system needs to keep adapting over time, as opposed to only once. In this work, we do consider several variants of pretraining among our baselines, and demonstrate that enabling continuous adaptation further improves their performance.

3. The Nevis'22 Benchmark

The **N**ever-**E**nding **VI**sual-classification **S**tream benchmark, dubbed Nevis'22, is a playground for research in never-ending learning. We start by summarizing its motivation and discussing how it was built and conclude with an analysis highlighting its key features.

3.1 Motivation

Our ultimate goal is to build a robust, efficient and autonomous never-ending learning system. For instance, we would like to provide the machine learning community with a never-ending learning model which can learn and integrate knowledge from whatever tasks the community

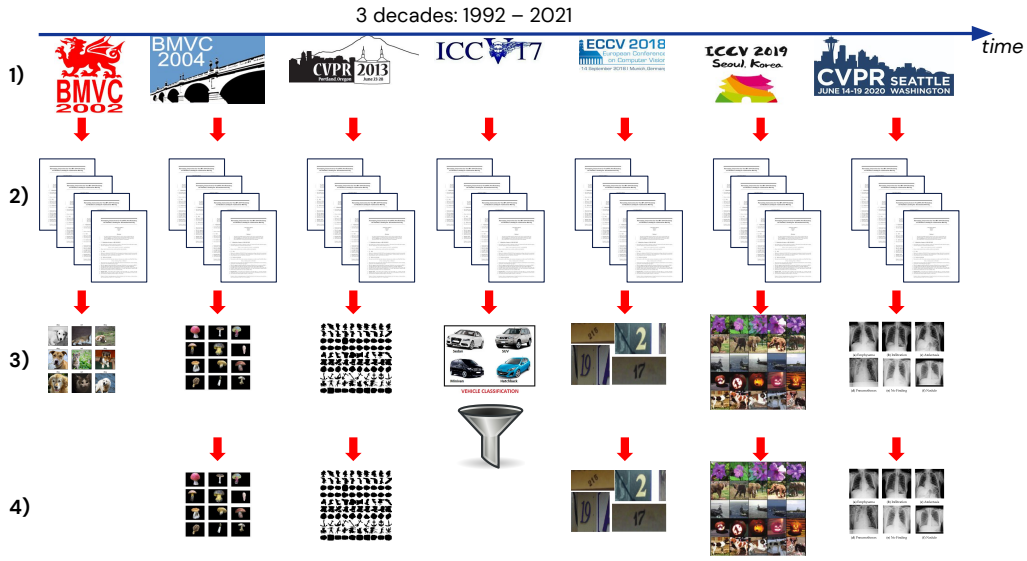


Figure 1: Illustration of the steps used to construct Nevis'22. First, we gathered and sorted chronologically proceedings of various computer vision conferences. Second, we sampled papers at random from each year. Third, for each paper we extracted the tasks used in the empirical validation. Finally, we filtered tasks. For instance, we retained only classification tasks that used publicly available data (see text for more detail).

considers at any point in time, and be used as a baseline for new methods being published. We do not make further assumptions on the task distribution, as this is generated by the community. We would like such never-ending learning system to never stop adapting, and to become more efficient over time, despite being exposed to more and more data and more and more complex tasks. Of course, there are many practical applications of such never-ending learning system, from auto-ml applications to robotics. In this work, we do not investigate what such a model could be, but focus on a benchmark which can be useful to develop such a model.

3.2 Stream Construction

The Nevis'22 benchmark is constructed according to four principles. 1) **Reproducibility:** This is an artifact for the research community, and therefore, data needs to be publicly available under permissive licenses. 2) **Simplicity:** The focus is on effective learning of *a sequence* of tasks, and therefore, each task must be well understood when taken in isolation. 3) **Agnostic task selection:** The selection of which tasks to include in the stream should not aim at favoring any particular approach. 4) **Scale:** The benchmark has an intermediate scale, useful for research in sequential learning. It is sufficiently large-scale to rule out approaches that do not scale gently with the amount of data. It is not too large to impede fast iteration of research ideas.

The protocol used in building Nevis'22 is illustrated in Fig. 1. We first gathered papers from leading computer vision conferences and workshops that host their proceedings publicly.

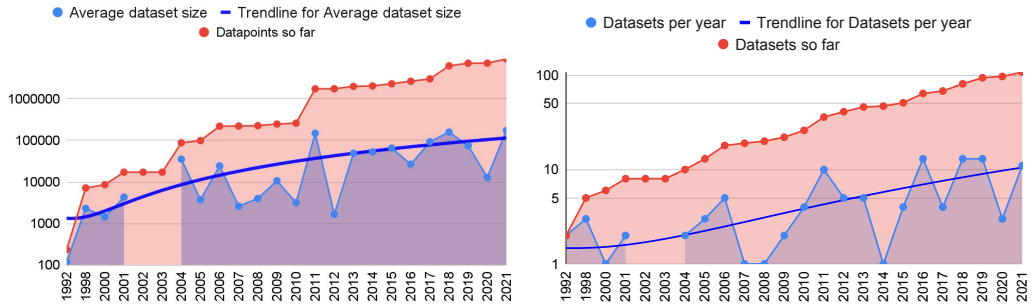


Figure 2: Left: Histogram of average dataset size each year, and cumulative number of datapoints in the stream. Right: Histogram of datasets per year. Most datasets have between 1000 and 20000 examples. The gap between 2001 and 2004 is due to duplicate removal, see section 3.2 for details. As expected, dataset sizes tends to increase over time. Note the log-scale of the plot.

We considered the British Machine Vision Conference (BMVC), the European Conference in Computer Vision (ECCV), the Computer Vision and Pattern Recognition conference (CVPR), the International Conference in Computer Vision (ICCV), ML4Health and Medical Imaging Workshops at NeurIPS. We sampled *uniformly at random* 90 papers each year (if available), from 1989 until 2021. Secondly, we manually extracted the tasks used for empirical validation. We filtered these tasks, retaining only i) classification tasks or tasks that can be mapped to classification, ii) tasks for which the corresponding dataset is publicly available, it is not deprecated and it has a permissive license for research purposes. Thirdly, we removed any duplicates that appear within a window of 10 years, retaining only the first instance. The rationale was to make the stream not too long or redundant, yet enabling the assessment of whether learning on subsequent instances of a dataset is faster or better. For example, using the heuristic above, we kept only the first instance of ImageNet from a paper published in 2011, removed all duplicate instances from years 2012 until 2020 and only retained a second instance from a paper published in 2021. Finally, the stream is the sequence of these tasks presented in the order in which they appeared in publications. We partition each dataset into three splits, namely training, validation, and test (see Sec. 4 for details). We remove any duplicate example to make splits and tasks fully disjoint. The full list of tasks is reported in Appendix L.

3.3 Stream Statistics

Nevis'22 is a stream composed of 106 image classification datasets totalling approximately 8 million images. There is a large diversity in data. For instance, the input *resolution* goes from 3×3 all the way to 2000×3000. Some datasets have fixed resolution, while for others each example has its own resolution.

The number of examples in each dataset also varies considerably, spanning four orders of magnitude, as it can be seen in Figure 2 left panel. Dataset size tend to increase over the years. Perhaps most importantly, Nevis'22 contains a large variety of domains, and yet within each domain there are enough datasets to support potentially beneficial transfer.

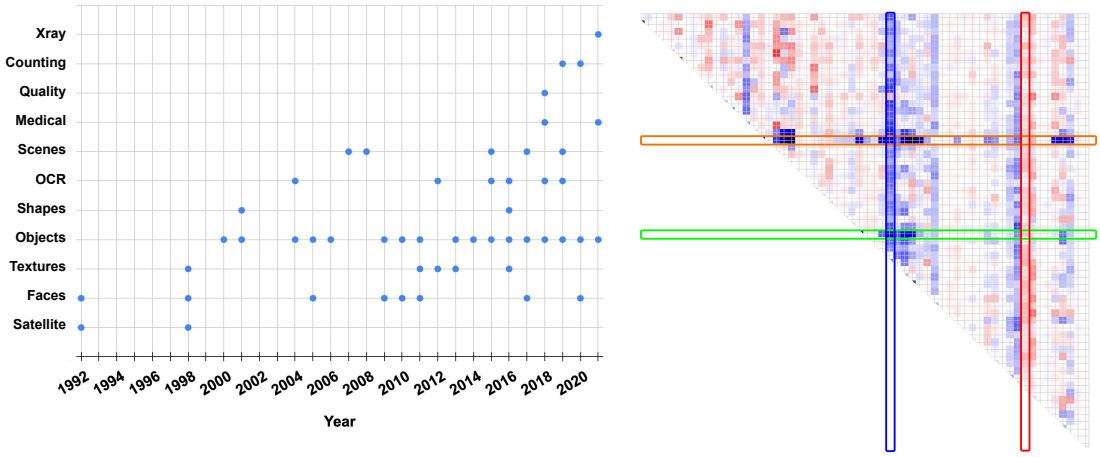


Figure 3: Left: Scatter plot showing the domains present in each year of the Nevis'22 stream. Each dot represents the presence of at least one dataset of a given domain in a certain year. The number of domains increases over time, and the popularity of domains vary with time. Right: Upper-triangular transfer matrix on a subset of tasks extracted from Nevis'22. The figure shows at position (i, j) the advantage of pretraining on task i before finetuning on task j, compared to learning task j from scratch. The upper triangular section shows only transfer from *tasks that had already occurred in the sequence*. Shade of blue indicate positive transfer (i.e. pretraining was useful), while shade of red indicate negative transfer. We notice that there exists tasks that are good for pretraining, leading to positive transfer to most future task (e.g., orange and green rectangles which correspond to ImageNet and SUN397 respectively), tasks that can transfer well from any other tasks (e.g., the blue rectangle corresponding to the Stanford Cars dataset) and tasks that do not transfer well from any other task (e.g., the red rectangle corresponding to the Mall dataset). See Appendix J for details.

The left plot of Figure 3 shows the major families of domain and their evolution over time. There are interesting patterns of non-stationarity, for instance with some domains appearing throughout the stream (e.g., object), while others being popular only in short time windows (e.g., satellite).

Note that such non-stationarity is a natural and desirable feature of Nevis'22, as it enables the development of models in a condition similar to deployment. Recall that at deployment time, the never-ending learning system might encounter tasks from entirely novel domains (for instance, around 2015 when the community started working on crowd counting or in 2020 when it started working on COVID-19 related tasks), as well as tasks with much more data than previously encountered (for instance, in 2009 when the ImageNet was introduced for the first time). Nevis'22 reproduces such natural stream of tasks, without making assumptions on the inner working of the never-ending learners.[3]

On the right side of Figure 3 we can also see how datasets relate to each other. The transfer matrix shows interesting structure, with both positive and negative transfer.

3. Notice that chronological order and uniform sampling of tasks might be undesirable If the objective were to merely maximize accuracy on a particular domain, for instance.

Algorithm 1 Training & Evaluation Protocol in Nevis'22

Initialization.
Meta-train stream: $\mathcal{S}^{\text{Tr}} = (\mathcal{T}_1, \ldots, \mathcal{T}_n)$
Meta-test stream: $\mathcal{S}^{\text{Ts}} = (\mathcal{T}_{n+1}, \ldots, \mathcal{T}_{n+m})$
Entire stream: $\mathcal{S} = \mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}}$
i-th task: $\mathcal{T}_i = (\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{val}}, \mathcal{D}_i^{\text{ts}})$.
Meta-learner: M.
# ***Meta-training phase:*** *Tuning M's hyper-parameters λ^M.*
repeat
    Designer chooses M's hyper-parameters λ^M.
    Initialize meta-learner state s_0 based on λ^M.
    for $\mathcal{T}_i \in \mathcal{S}^{\text{Tr}}$ **do**
        $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{val}} \leftarrow \mathcal{T}_i$
        $P_i, \text{FLOP}_i \leftarrow$ Using $s_{i-1}, \mathcal{D}_i^{\text{tr}}$ and $\lambda^M : M$ performs h.p. search and trains P_i.
        $s_i \leftarrow M$ updates its state using $\mathcal{D}_i^{\text{tr}}, s_{i-1}$ and λ^M.
        $e_i \leftarrow$ error rate of P_i on $\mathcal{D}_i^{\text{val}}$
    end for
    $\mathcal{E}(\mathcal{S}^{\text{Tr}}) = \sum_{i=1}^{n} e_i$
    $\text{cFLOP}(\mathcal{S}^{\text{Tr}}) = \sum_{i=1}^{n} \text{FLOP}_i$
until Designer is happy with choice of λ^M based on M's performance.
Meta-test phase: *Evaluating M.*
Initialize meta-learner state s_0 based on λ^M.
for $\mathcal{T}_j \in \mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}}$ **do**
    $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{ts}} \leftarrow \mathcal{T}_j$
    $P_j, \text{FLOP}_j \leftarrow$ Using $s_{j-1}, \mathcal{D}_j^{\text{tr}}$ and $\lambda^M : M$ performs h.p. search and trains P_j.
    $s_j \leftarrow M$ updates its state using $\mathcal{D}_j^{\text{tr}}, s_{j-1}$ and λ^M.
    $e_j \leftarrow$ error rate of P_j on $\mathcal{D}_j^{\text{ts}}$
end for
$\mathcal{E}(\mathcal{S}^{\text{Ts}}) = \sum_{i=n+1}^{n+m} e_i$.
$\text{cFLOP}(\mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}}) = \sum_{i=1}^{n+m} \text{FLOP}_i$.
Return and report
Average error rate of meta-test stream: $\mathcal{E}(\mathcal{S}^{\text{Ts}})$
Cumulative FLOPs of entire stream: $\text{cFLOP}(\mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}})$

The scale, diversity and agnostic selection used in the construction process are the defining elements of Nevis'22, compared to existing benchmarks that might satisfy some of these desirable axes but, to the best of our knowledge, not all of them.

4. Learning & Evaluation Protocol

We recall that the Nevis'22 benchmark operates on a sequential stream of diverse tasks; and that learners are evaluated on their ability to efficiently generalize what they have learned early on in the stream to tasks that appear later. An important aspect of this setting is

that learners are not permitted to use future observations to influence their behavior on a given task. Most notably, this condition is also intended to apply to the selection of the hyper-parameters used by learners. In particular, learners are not permitted to select hyper-parameters based on metrics computed on runs over the full stream, since this is equivalent to using information about future tasks in the stream to influence the present. This decision has been made to encourage algorithm designers to build robust learning systems that can truly adapt to changing distributions automatically, rather than through careful human-driven hyper-parameter tuning. These requirements necessitate a rigorous evaluation protocol that supports both the iteration and development of learners (including their meta-learning components), and also meaningful comparisons of learner implementations once they have been tuned. In this section, we outline the strategies we have adopted to support these requirements.

In all generality, we assume that there exists a "meta-learner" M that is in charge of instantiating a predictor P_i for the i-th task. M is responsible for determining any hyper-parameters (such as learning rate and label smoothing values) needed to construct P_i, and also for possibly leveraging the results of previously observed tasks, if doing so could enable the learner to more efficiently solve the current task. For instance, M could initialize the parameters of P_i from the parameters of a network trained on a related previous task. Furthermore, M itself might have hyper-parameters; for instance, they could be the range of values considered by random hyper-parameter search, or the choice of transfer learning method. How shall we cross-validate the meta-learner M and each predictor P_i? How should we account for the compute used by both M and P_i? To answer these questions, we propose the training and evaluation protocol described in Algorithm 1.

Since we are interested in assessing generalization to future tasks, we divide Nevis'22 into two sub-streams (line 2 and 3): the "meta-train stream", denoted by $\mathcal{S}^{\text{Tr}}$, and "meta-test stream", denoted by $\mathcal{S}^{\text{Ts}}$. $\mathcal{S}^{\text{Tr}}$ comprises the tasks contained in the first 27 years for a total of 79 tasks, $\mathcal{S}^{\text{Ts}}$ contains the 27 tasks from the last 3 years. The choice of how many tasks to include in $\mathcal{S}^{\text{Ts}}$ versus $\mathcal{S}^{\text{Tr}}$ strikes a good trade-off between: a) having a sufficient number of tasks that can be used for development and b) having enough tasks to assess generalization at meta-testing. The last 27 tasks which make $\mathcal{S}^{\text{Ts}}$, are listed in tab. L. Among the 27 datasets, there are some duplicate from metatrain (e.g., ImageNet, Oxford Flowers 102) and datasets from various domains like OCR, object, counting, scene understanding, medical, etc. Four datasets are from a new sub-domain, medical COVID-19 x-ray. We therefore believe $\mathcal{S}^{\text{Ts}}$ provides a nice coverage of the scenarios encountered by a never-ending learning system.

Finally, each task $\mathcal{T}_i$ consists of three datasets, namely training $\mathcal{D}_i^{\text{tr}}$, validation $\mathcal{D}_i^{\text{val}}$, and test $\mathcal{D}_i^{\text{ts}}$. Next we explain how these streams and datasets splits are used.

4.1 Meta-training Phase

During the meta-training phase (lines 8 to 19), a designer (effectively, a *meta* meta-learner) might run the meta-learner M multiple times over $\mathcal{S}^{\text{Tr}}$ to tune M's hyper-parameters λ^M; for instance, this might include choosing neural network architectures, optimizers, data-augmentation strategies and intialization parameters. It is up to the designer to decide which configurations to try next (line 9), and when to stop the search (line 19). At every such iteration, M sweeps over the tasks (or any subset thereof) of $\mathcal{S}^{\text{Tr}}$. It first extracts the task

specific training and validation set (line 12). Then it uses the training set $\mathcal{D}_i^{\text{tr}}$ to produce a predictor P_i for the i-th task. This process typically involves some form of task-level search over P_i's hyper-parameters. In order to support this, Nevis'22 provides a default decomposition of $\mathcal{D}_i^{\text{tr}}$ into two sets, one used for actual training of P_i and the other used for task-level cross validation. However, it is up to the meta-learner to decide whether to use this or other ways to partition $\mathcal{D}_i^{\text{tr}}$ to better support P_i's hyper-parameter search. The result of this step is not only P_i, a predictor for task i, but also the total number of floating point operations used during this training and hyper-parameter search process, denoted by FLOP_i.

Notice that M uses a certain configuration of its own hyper-parameters λ^M and an internal state s_{i-1} to find P_i. Examples of λ^M could be the range of learning rate values used during the actual random search of P_i's learning rate. The state s_i instead is what represents the knowledge accrued up to the i-th step. This can be an empty set if M instantiates independently learners to tasks. It could also consists of the set of parameters used by P_j for $j < i$, supporting various kinds of finetuning strategies from models trained on previously encountered tasks. This state is updated by M in line 14. In the previous example, this merely consists of adding an additional parameter vector to a pre-existing set of parameter vectors, one for each previous task. Finally, P_i is evaluated on $\mathcal{D}_i^{\text{val}}$ (line 15) in terms of error, as follows:

$$e_i := \begin{cases} 1 - \text{acc}_i & \text{if i is a single-label task} \\ 1 - \text{mAP}_i & \text{if i is a multi-label task,} \end{cases} \tag{1}$$

where acc_i is the average accuracy on the task i; mAP_i is the mean average precision on the task i.

Ultimately, these task level metrics, (e_i, FLOP_i) are aggregated at the stream level via averaging or sum. Given a stream $\mathcal{S}$ with K tasks, we define:

$$\mathcal{E}(\mathcal{S}) := \frac{1}{K} \sum_{i=1}^{K} e_i \tag{2}$$

$$\text{cFLOP}(\mathcal{S}) := \sum_{i=1}^{K} \text{FLOP}_i \tag{3}$$

where we denote the average error rate with $\mathcal{E}$ and the cumulative floating point operations with cFLOP. By varying hyper-parameters such as the number of gradient steps used, the size of the architecture or the number of trials used during hyper-parameter search, M will strike different trade-offs between average error rate and total compute. The search process over λ^M will aim at pushing the Pareto front of these points. In our work (as it is standard practice), the process of searching over λ^M requires a human in the loop to determine the best meta-learner's configuration to explore next [4].

Note that one may choose to train on all tasks at once, or to pick only the largest one. Yet the meta-training loop is still used to find λ^M. Researchers are however free to explore

4. We focus on the Pareto front because we are faced with multi-objective optimization: for each meta-learner M and (meta-learner) hyperparameter setting we obtain a predictive performance $\mathcal{E}$ and cFLOP. The Pareto front illustrates the best attainable performance for each compute budget.

other ways to cross-validate M, for instance by introducing a meta-validation stream. In this work, we opted for simplicity, but we leave open the question of how to best cross-validate M and hope that Nevis'22 can be useful also to address this research question.

4.2 Meta-testing Phase

Once λ^M is determined, we evaluate M (lines 21 to 30). The process follows the same steps discussed previously, with a few exceptions. First, M must do a pass over the *entire* stream, $\mathcal{S} = \mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}}$. Notice that at step i M cannot access any task $\mathcal{T}_j$ with $j > i$; in particular, since there is no outer loop over M's hyper-parameters, tasks in $\mathcal{S}^{\text{Ts}}$ are observed once and only once in sequence. During training on task $\mathcal{T}_i$, P_i can do several epochs over $\mathcal{D}_i^{\text{tr}}$, but M cannot revisit $\mathcal{T}_i$ twice because otherwise we would not be able to assess generalization to unseen tasks of $\mathcal{S}^{\text{Ts}}$. The second difference is that P_i is evaluated on $\mathcal{D}_i^{\text{ts}}$. Finally, the average error rate is calculated only using tasks belonging to $\mathcal{S}^{\text{Ts}}$. The rationale is to remove tasks over which we did any kind of cross-validation to prevent overfitting. However, we still account for the cost of development of M by including the FLOPS used while learning on $\mathcal{S}^{\text{Tr}}$ (although for the sake of simplicity, we do not consider how many times $\mathcal{S}^{\text{Tr}}$ was visited during the meta-training phase).[5]

Ultimately a given method will yield a tuple, $(\mathcal{E}, \text{cFLOP})$, where $\mathcal{E}$ is the average error over $\mathcal{S}^{\text{Ts}}$ and cFLOP is the cumulative compute over the whole stream $\mathcal{S}$. The best methods will be the ones at the Pareto front, delivering the lowest $\mathcal{E}$ for a given total compute budget. A clever method could early stop on tasks whenever its gains in terms of error rate are too marginal, and use the saved compute on future tasks that do require more compute to achieve lower error rate.[6]

4.3 Computational requirements and the SHORT stream

Depending on algorithm, hyperparameters and available hardware, a single run on the benchmark can often take multiple days even on machines equipped with 16 A100 NVIDIA GPUs. In Appendix C we report experiments with cheaper computational budgets that could run for a few days on a single GPU device.

To further facilitate quick experimentation and to ensure researchers with limited access to compute resources can contribute, we derive a SHORT stream by randomly selecting only two datasets per year from the original list of publications; but otherwise following the same stream creation procedure. We obtain a stream with 24 tasks in total. Table M lists the datasets and their chronological order. The majority of learning algorithms described in this study finish in under 24h on this stream when running on 16 A100 GPUs. We used SHORT for all our initial experiments.

5. Using a separate set of test tasks, $\mathcal{S}^{\text{Ts}}$, and test splits for the datasets, $\mathcal{D}^{\text{ts}}$, might seem overly zealous. This was however useful to assess potential overfitting to $\mathcal{S}^{\text{Tr}}$, and for some of our ablations which analyzed the full stream $\mathcal{S} = \mathcal{S}^{\text{Tr}} + \mathcal{S}^{\text{Ts}}$. For instance, the ablation by domain of Sec. 5.7 required the analysis on the full stream.
6. While we recommend to rank methods by reporting pareto-fronts of compute versus error rate, section 5.7 reports additional metrics which provide finer grained understanding of strengths and weakness of different methods; for instance, we consider slicing results by domain, dataset size and we also report forward transfer on datasets appearing more than once.

4.4 Codebase

An open source implementation of the Nevis'22 benchmark has been made available at `https://github.com/deepmind/dm_nevis`. This repository implements the training and evaluation protocol described in this section, downloads and prepares the data, and enables other researchers to reproduce the main results we obtained with our baselines.

The implementation has been designed to be modular, compartmentalizing data processing, handling of the stream, learners and metrics. In particular, no knowledge of the particular stream or metrics is needed in order to implement a new learner. Moreover, the learner interface is minimal, requiring only the implementation of functions that initialize, train, and compute predictions. This may be implemented using any appropriate Python based machine learning library such as JAX or PyTorch.

5. Experiments

In this section we describe the baseline approaches and results we obtained on the Nevis'22 stream. In Sec. 5.7 we conclude with ablations showing what factors contribute to the performance of current approaches, and how the diversity and scale of Nevis'22 enable better assessment of life-long learners.

5.1 Preprocessing and Data Augmentation

The tasks in Nevis'22 have images spanning a wide range of different resolutions. Even within a task the resolution may vary from image to image. For our baselines, we adopted a two-part strategy which favored simplicity over performance. During training, images are randomly resized and cropped to a fixed resolution of 64×64 pixels, and then left-right flipped with probability 0.5. During evaluation, we take the central square crop of $\min(w, h) \times \min(w, h)$, where w is the width of the image and h is its height, and resize it to 64×64 pixels with no additional augmentation.

Note that this strategy is clearly sub-optimal for tasks involving fine details, such as crowd counting. Nonetheless, this choice simplifies the design of architectures. More details can be found in the sensitivity analysis of Fig. 9.

5.2 Architecture

Unless otherwise stated, our baselines use the ResNet34 backbone tailored for low resolution images (He et al., 2016, Sec. 4.2), since all images are resized to 64×64 pixels. Each task is assigned a task-specific *head* mapping backbone features to output logits.

5.3 Meta-Learning

Each of our baselines includes a *meta-learner* that selects the task-specific hyper-parameters used during training of the actual predictor. For the sake of speeding up tuning of the meta-learner, we performed stream-level cross-validation on the SHORT stream, containing 24 tasks only. We tuned the choice of the architecture and the ranges used by the hyper-parameter search. In particular, we identified a set of hyper-parameters that are robust enough to be kept fixed throughout the learning experience on the stream: 1) cosine learning

Baseline	initialization	training data	h.p. search
Independent (Indep)	random	current task	random search
Fine-tuning (FT)	from a previous task	current task	random search
Multi-tasking (MT)	from a previous task	multiple tasks	random search
Pre-training (PT)	from pre-trained model	current task	random search
PT + FT	from a previous task or a pretrained model	current task	random search
Bayesian hyperparameter optimization (BHPO)	random	current task	Gaussian process with upper confidence bound (GP-UCB)

Table 4: Differentiating baselines across three axes: 1) How the parameter's of the predictor are initialized, 2) What data is used to train the predictor and 3) How hyper-parameter search of each predictor is conducted.

rate scheduling with warm-up phase proportional to the number of gradients updates in conjunction with SGD with Nesterov momentum (set to 0.9, with a weight decay of 0.0001), 2) data augmentation consisting of random cropping and flipping, and 3) a heuristic to set the batch size as a function of the dataset size,

$$b = \min\left(B, \max\left(16, 2^{\lfloor \log_2 p \cdot D \rfloor}\right)\right), \tag{4}$$

where B is the maximum batch size, D is the dataset size, and p is a constant set to 0.0025 in our experiments.

5.4 Learning

All baselines we have considered in our empirical evaluation yield one predictor per task, without any parameter sharing across tasks. This is the simplest setting which corresponds to the most widely used design choice in practice. There are three independent factors that are used to create each baselines: 1) The choice of initialization, 2) The choice of which data is used to train a given predictor, and 3) The choice of the algorithm used to search in hyper-parameter space. The particular combination of three factors determines the meta-learner M described in Sec. 4. In this work we have considered the six most widely used combinations of these three factors, as summarized in Tab. 4, namely:

1. **Independent (Indep):** The meta-learner initializes the parameters of the predictor for each task at *random*, trains using data for the *current* task only, and searches over hyper-parameters using *random search*. This is the the most naïve baseline. It is the

reference that any other method should beat, as it is the simplest method that does not support any form of transfer learning (the state of the meta-learner s_k is null for all k).

2. **Finetuning (FT):** The meta-learner initializes the parameters of the predictor from the parameters of a network trained on *a previous task*, trains using data for the current task only, and searches over hyper-parameters using random search. In this case the state of the meta-learner s_k consists of the union of the model parameters trained on all tasks observed so far, from 1 till $k-1$. Knowledge for this learner is the set of model parameters, which correspond to one expert per observed task.

   We have considered various criteria to select the previous task from which to finetune: 1) temporal proximity by taking the most recent task (FT-prev), 2) task relatedness by taking the most related past task. For the latter, we have been using as a proxy of task relatedness the performance of a k-nearest neighbor classifier in the feature space produced by the previous predictors, using as training and validation data a small subset of $\mathcal{D}^{tr}$ (up to 10000 and 5000 images, respectively). In other words, we use the previous predictors as feature extractors to encode data from the current task, similarly to Veniat et al. (2021). We have two versions of this. An offline or static version (FT-s) where the features are computed using pretrained independent predictors as in Indep above, and a dynamic version (FT-d) where features are computed online using the actual predictors trained so far (which could have been finetuned themselves on other tasks). Fig. 18 in Appendix shows an actual example of such learned chain of finetuned models.

3. **Multitasking (MT):** The meta-learner initializes the parameters of the predictor using the same approach as in FT-d, trains using data of *both the current task and some previous tasks*, and searches over hyper-parameters using random search. Both the selection criterion for what previous task to take for parameter initialization and what previous (auxiliary) tasks to take for additional training data are based on task relatedness using the same k-nearest neighbor classifier score described in the FT-d baseline above. Unlike FT, the multitask baseline can take $k \geq 1$ most related auxiliary tasks for additional training data. During training the network is trained in a multitask fashion, weighing the losses of the auxiliary tasks by a single scalar hyper-parameter λ. This hyper-parameter is subject to hyper-parameter search by the meta-learner, and it sets the relative importance of the auxiliary tasks against the current task. In a single training step, the gradients are accumulated across the mini-batches of current and k auxiliary tasks. At test time, the classification heads of the auxiliary tasks are disregarded. With reference to Algorithm 1, the state s_k of the meta-learner M consists of the set of datasets and predictors trained so far. In this case, knowledge is represented both as the set of model parameters of already observed tasks, as well as the set of datasets encountered so far.

4. **Pre-training (PT):** The meta-learner initializes the parameters from a *pretrained* model, trains using data for the current task only, and searches over hyper-parameters using random search. This is a special case of FT, where all task predictors are finetuned

from exactly the same pretrained model. Note that there is no form of knowledge accumulation for this baseline.

We have considered two pretrained models from which to finetune: 1) A ResNet34 pretrained on ImageNet by supervised learning (PT-ISup) and 2) A Normalizer-Free network (NFNet-F0) (Brock et al., 2021) pretrained on two very large external datasets: ALIGN and LTIP (Alayrac et al., 2022) using CLIP (PT-ext) (Radford et al., 2021). In this case, the state s_k of the meta-learner M is constant over time, as it merely contains the fixed set of pre-trained parameters.

5) Fine-Tuning with Pre-training (PT + FT): This variant is exactly the same as FT-d, except that the set of parameters avaiable for finetuning includes not only the parameters of networks trained on previous tasks but also the pretrained model used by PT-ext.

6) Bayesian Hyper-Parameter Optimization (BHPO): The meta-learner initializes the parameters of the predictor at random, trains using data for the current task only, and searches over hyper-parameters using a *Gaussian Process* to estimate the function value (expected loss at convergence). Instead of running a search in parallel over a set of hyper-parameter configurations like in random search, BHPO runs the search in sequence, using the Upper Confidence Bound acquisition function (Srinivas et al., 2010) provided by Google Vizier (Golovin et al., 2017) to select the next configuration to search over.

5.5 Experimental Setup.

The search space of Indep, FT, PT and PT+FT consists of initial learning rate and label smoothing, which means that for each task we search over the values of these two hyper-parameters. MT adds to the search space also λ, the weight on the auxiliary tasks. BHPO adds five additional hyper-parameters compared to Indep, which include the choice of learning rate schedule (cosine learning rate, piece-wise constant decay), batch size, choice of architecture (VGG, ResNet34), choice of the two data augmentations (random cropping and flipping).

To vary the compute budget and study the Pareto front of average error rate versus compute, and unless otherwise stated, we vary the number of hyper-parameter configurations over which the meta-learner searches over at each task (ranging from 2 to 32), and the total number of weight updates (ranging from 10000 to 100000). Note that different combinations of number of updates and trials per task can lead to the same computational budget, but different performance.

In the following section we will report results for a total compute budget in the range between 10^{18} and 10^{21}, aiming at models that achieve competitive final performance (for the chosen 64×64 resolution) on landmark datasets. For instance, the basic Indep baseline achieves 4.2% error rate on SVHN, 0.7% on MNIST, 6.8% on CIFAR10 and 34% on ImageNet. In Appendix C, we will report results using smaller computational budgets (and hence worse final error rate) for users that have more limited computational resources at their disposal.

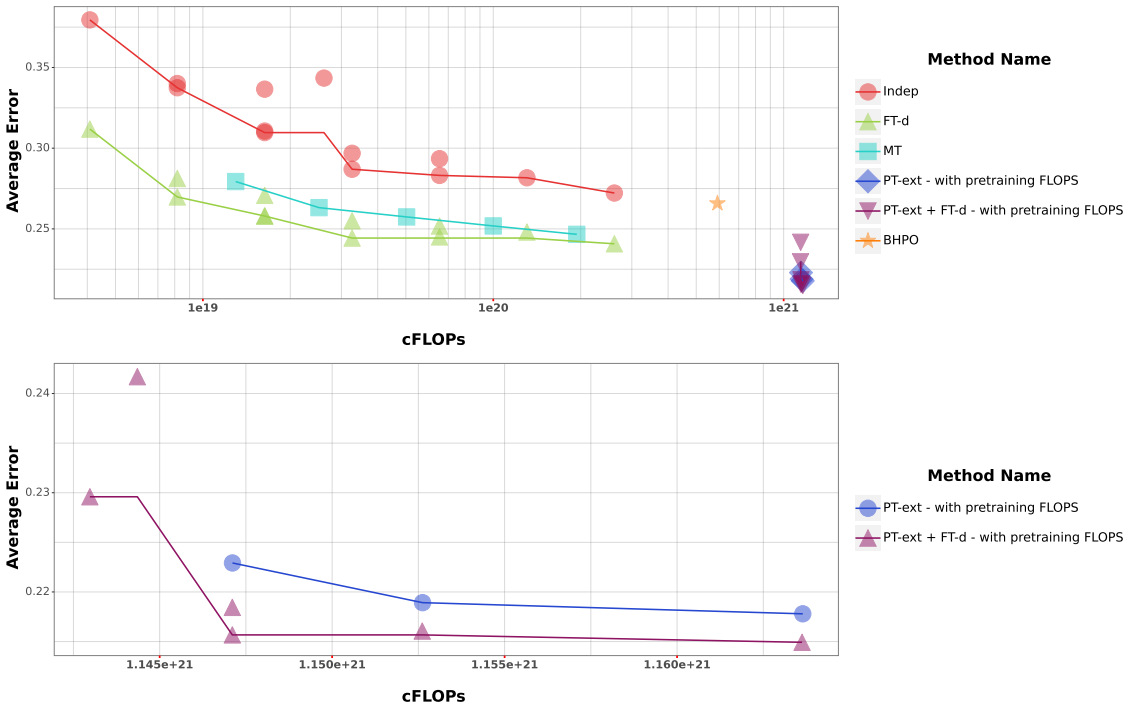


Figure 4: **Pareto fronts**: Each marker shows the average error rate on $\mathcal{S}^{\text{Ts}}$ and the total cFLOP on the entire stream (see Sec. 4). Since there are 106 tasks, if each task required 16 hyper-parameter configurations, a marker is the result of 1696 experiments. Pareto fronts are created by varying the number of hyper-parameter configurations and the number of gradients steps used to train on each task. The top panel shows a selected baseline from each of the baseline families described in Sec. 5.4, and the bottom panel zooms into the pretraining baselines using external data.

5.6 Findings

In this section we report the main results we obtained by applying the previously described baselines to Nevis'22. A priori, common wisdom would suggest that methods that perform multi-task using all available data would perform the best, while methods that rely on sequential finetuning to grossly underperform (Ash and Adams, 2020). We would also expect methods based on pretraining to perform the best, but to gain little if anything by combining with other forms of transfer learning as their representations are potentially already general enough. We would also expect Indep to work more poorly on smaller datasets, and that no method is best across the entire spectrum of compute budget. Our results show that not all these intuitions find empirical support in Nevis'22.

We start by reporting the pareto fronts of average error rate versus compute in Fig. 4. In this figure, we show a selected baseline from each of the families described in Sec. 5.4. More results are reported in the Appendix (Sec. E).

1. We observe that all methods perform better than learning from scratch (Indep). This shows that indeed there is rich shared structure across tasks in Nevis'22 which supports

beneficial transfer. Note that a run with 3 different seeds of the independent baseline (at the second highest cost) yields a standard deviation of 0.003, suggesting that performance gaps between baselines are significant except for FT-d and MT at the highest computational budget.

2. BHPO reduces the error compared to Indep, but at the cost of more compute. Overall, BHPO does not improve the pareto front. Given that hyper-parameter search substantially impacts cFLOP (typically by a factor of ten or more), we surmise that there could be more clever and efficient ways to explore the hyper-parameter space which could lead to a better performance-compute trade-off.

3. There is a rather large gap of about 5% absolute error between the best and the worst method, namely Indep and FT-d if we restrict to training data from Nevis'22. Given the simplicity of FT-d in terms of its approach to transfer learning and its naïve use of compute (random hyper-parameter search and no early stopping), we would expect future approaches to further reduce both compute and error rate.

4. The performance of FT-d suggests that sequential finetuning, even when applied on chains longer than 10 steps, as it can be appreciated in Fig. 18 of Appendix, works remarkably well.

5. The choice of what to finetune from matters, as FT-prev is significantly worse than FT-d as shown in the Appendix (Sec. E.1). Therefore, there could be improvements by using better approaches to estimate task relatedness.

6. MT works well but the additional compute spent on relearning representations on past tasks does not compensate for the improvements in generalization. Overall, FT-d strikes a better trade-off than MT. Note that the experiments in the main paper are reported with $k = 1$. A more detailed ablation on this hyperparameter is provided in the Appendix (Sec. G).

7. As expected, pretraining improves the performance significantly. Starting with a pretrained network (PT-ext) leads to a significantly lower average error. Notice that PT-ext leverages both a much larger amount of external data and a more powerful architecture. In the Appendix E.2 we also show that pretraining the same architecture used for the other baselines on ImageNet (PT-ISup) reduces the gap between Indep and the best performing baselines (FT-d and MT).

8. More surprisingly, using FT-d with a pretrained network (PT-ext + FT-d) lowers the average error further, see bottom plot of Fig. 4. This demonstrates that leveraging the structure in the stream can improve the already general representations that the pretrained model provides, which opens a new avenue of research on large-scale models that continuously adapt over time. More details on the structure that FT-d discovers starting from the pretrained model are provided in the Appendix E.1.2.

In order to better understand how methods fare on each task, we also present a regret-like plot in Fig. 5. This shows the cumulative error over time relative to Indep, picking hyper-parameters such that all methods use roughly the same amount of compute. When

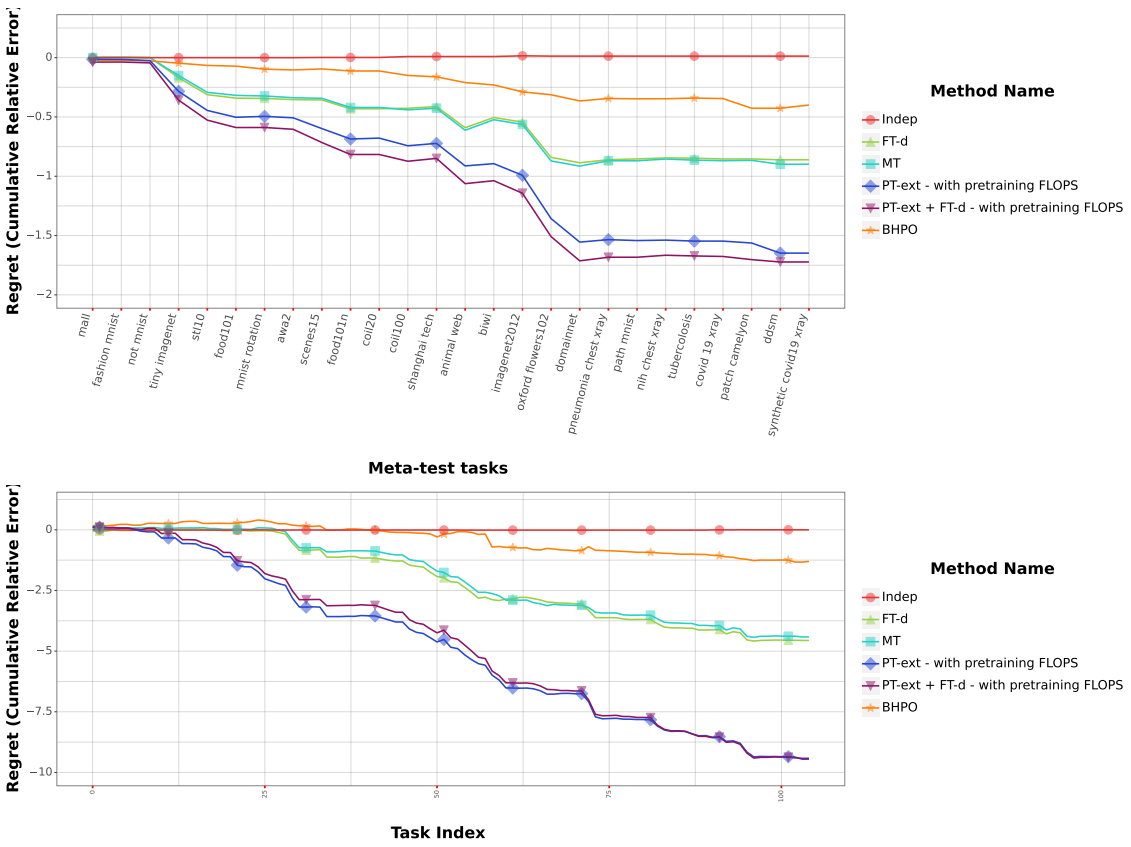


Figure 5: **Regret plots**: Cumulative error rate relative to Indep. on $\mathcal{S}^{\text{Ts}}$ (top) and on the full stream (bottom).

the curve is horizontal it means that a method performs comparably to Indep. When the slope is negative it means that it outperforms Indep, and vice versa. We observe that no method, including PT variants, transfers well to datasets in the OCR and medical domains. In particular, all regret curves are nearly horizontal over the last nine datasets, which are mostly datasets in a new x-ray domain (Covid-19 related classification tasks from 2021). This shows a clear limitation of current approaches which are not yet capable to a) transfer well to minor domains, and b) accrue knowledge over time, as many of these 9 last tasks are closely related to each other.

Finally, the regret plot over the entire stream in the bottom of Fig. 5 shows that overall methods provide a linear improvement over the Indep baseline. FT-d starts flat as expected (since initially there is nothing to transfer) and later on exhibits a linear gain. However, no method improves over time in the second part of the stream. In other words, none of the baselines we tried was capable to become more accurate as it receives more data and it makes new learning experiences. While this is expected for PT which cannot accrue knowledge by construction, we surmise there could be a more clever variant of FT that actually improves over time.

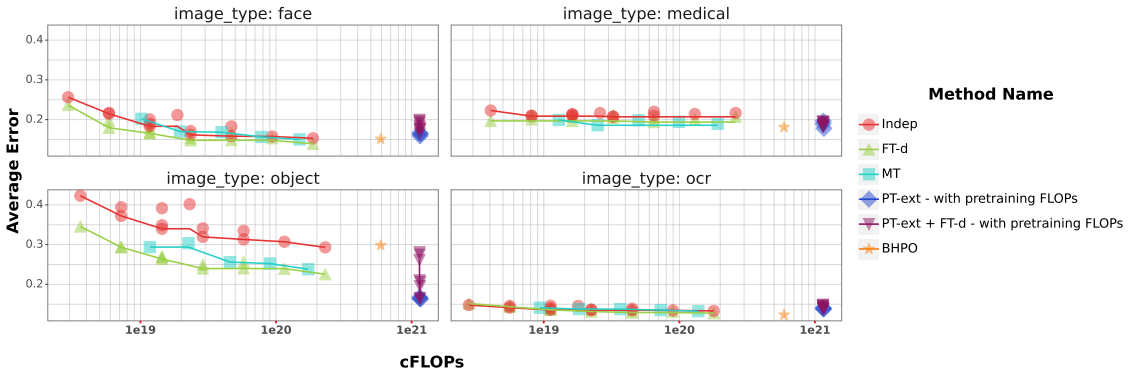


Figure 6: Analysis by domain. Each sub-plot corresponds to a different domain.

Overall, these findings indicate that NEVIS'22 is a good playground for research in never-ending learning. Methods that transfer do significantly better than methods that do not, and yet there seems to be ample room for improvement over the current set of baselines. Unlike our expectations, methods based on multi-tasking did not yield better trade-offs, they might achieve a lower error rate but this does not compensate for the increase in the amount of compute. Instead, sequential FT has worked remarkably well despite the simplicity of the heuristic used to determine task relatedness. This observation still holds when starting from a large pretrained model, opening the question of how to best accumulate knowledge in foundational models. Finally, we have reported more results using BHPO in Appendix H. These show that current BHPO is effective only when increasing the search space. In this setting, BHPO does find better hyper-parameter configurations than random search but the gains are currently too limited relative to the additional compute required.

5.7 Ablations

In this section we further analyze NEVIS'22, studying how results vary by domain, dataset size, task ordering, etc. The goal is to understand which factors affect the performance of the baselines the most, and ultimately, which unique features NEVIS'22 has to offer relative to existing benchmarks.

5.8 Image Domain

In this experiment, we take baselines which have been trained on the entire stream, and instead of evaluating on all tasks of $\mathcal{S}^{\text{Ts}}$, we evaluate on all tasks of $\mathcal{S}^{\text{Tr}} \cup \mathcal{S}^{\text{Ts}}$ but filtering tasks by their domain. For the evaluation we use the test split of each task. We report results on 4 representative domains, namely OCR, medical, face and object. Note that in methods like FT, networks that are trained on a dataset of a certain domain, could be finetuned from a task belonging to a different domain. Moreover, results across domains are not directly comparable since each domain has its own set of tasks. Results are shown in Figure 6. We observe a strong dependence on the domain type. The gains over Indep brought by baselines that allow transfer learning are minimal in OCR but substantial for object, for instance. Even more interestingly, the ranking of the various baselines is domain dependent, and there

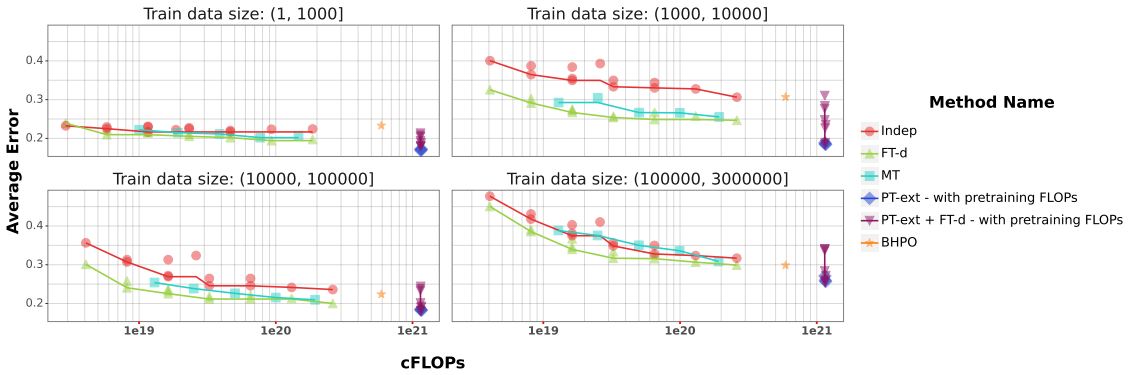


Figure 7: Analysis by dataset size. Each subplot contains the evaluation on datasets of the size indicated on the top.

is no winner across all domains. For instance, PT is the best method in the medical domain but the worst in OCR. We conjecture that the reason could be that OCR has relatively large datasets, but it is perhaps the most distant domain relative to the domain used for pretraining. Notice how these insights have been enabled thanks to the richness of domains in Nevis'22.

5.9 Dataset size

We now study how performance correlates with dataset size. We use the same methodology used in the previous analysis by domain, but aggregate test results filtering by the size of the datasets. We have defined four groups of increasing size of the training set, namely datasets with a training set with fewer than 1,000 examples, datasets with a number of training examples between 1,000 and 10,000, datasets with a number of training examples between 10,000 and 100,000 and datasets with more than 100,000 examples. Results are shown in Fig. 7. Unsurprisingly, Indep becomes competitive on very large datasets, but surprisingly, methods that support transfer learning (e.g., FT) do not gain much over Indep on very small datasets. The gains are more significant on datasets of intermediate size. Overall, adapting to small datasets is still a challenge for the baselines we have considered. Once again, Nevis'22 has enabled this analysis thanks to its diversity of dataset sizes.

5.10 Image resolution

With a methodology similar to the one used in the previous experiments, we again train on the full stream (using the default fixed resolution of 64×64 pixels), evaluate on the test split of each task but report metrics by selecting only tasks with average image resolution within a certain range. Fig. 8 shows that the error rate on datasets with smaller resolution is very low, and on those datasets methods that transfer work comparably to Indep. As the resolution increases we observe a remarkable improvement of FT, PT and MT over Indep. Finally, while we cannot directly compare results across various image resolutions (since each contain a different subset of tasks), we notice that the average error rate on datasets of larger resolution images is much bigger, suggesting that downsizing the resolution to

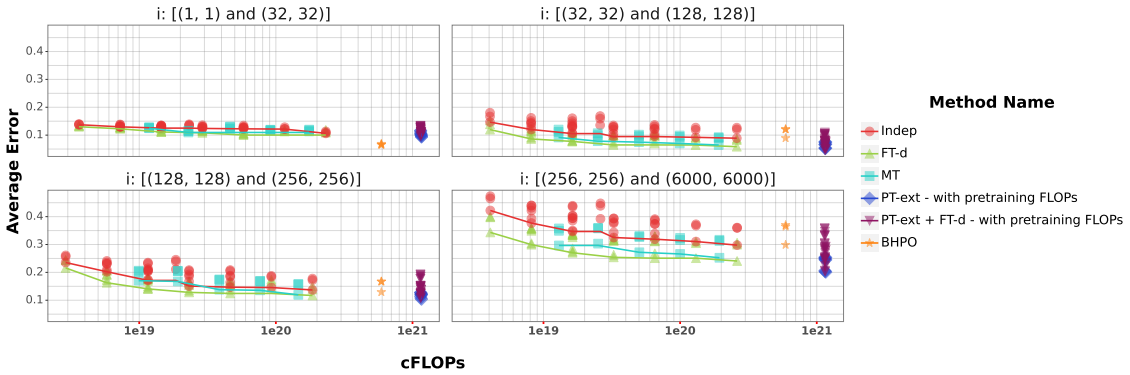


Figure 8: Analysis by the average image resolution of each task. We train on images resized to 64×64 pixels but evaluate by selecting tasks with original image resolution within the specified range.

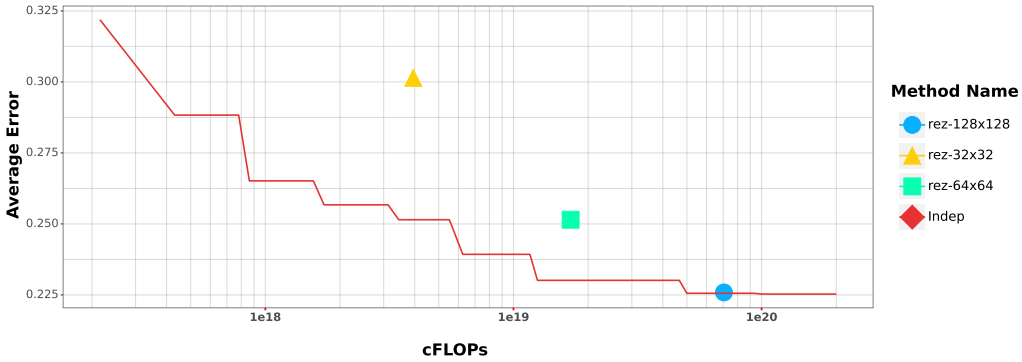


Figure 9: Performance of Indep when varying the image resolution training and evaluation using ResNet34 with a maximum batch size of 128, 50,000 gradient updates, and 16 trials per task. The red line is the Pareto front of the default Indep baseline trained on images of size 64×64 using different models, a maximum batch size of 512, and various combinations of number of updates and trials per task.

only 64×64 pixels might deteriorate performance on such tasks, and that architectures that handle variable resolution images might strike better trade-offs. Alternatively, we also studied how the performance of Indep changes as we vary the choice of the input image resolution used during training and evaluation of each task. This was chosen to be 64×64 by default in our previous experiments. For the experiment of Fig. 9, we tried two other spatial resolutions, namely 32×32 and 128×128. Note that this experiment is conducted on the short version of the stream. The red line represents the Pareto front of the Indep baseline, trained on the default image resolution (64×64) and default maximum batch size (512), and varying the model architecture, the number of updates and the number of hyper-parameter configurations over which we search for each task. This frontier is provided as a reference. We can see that varying the image resolution is yet another way to trade-off error rate versus compute. For example, it is noticeable that the point at 128×128 resolution intersects with

Indep Pareto frontier. It would be interesting to complete these curves by varying the image resolution or other factors like the architecture size and study if we can improve the Pareto fronts, but we leave this exploration for future works.

5.11 Task ordering

In this experiment we study whether the performance of baselines is affected by the ordering of the tasks. For this purpose, we create two new variants of Nevis'22, one where we pick a different shuffling of the datasets within a year (note that in Nevis'22 this is already arbitrary), and one where we shuffle the order of the datasets across the entire stream, $\mathcal{S}^{\text{Tr}} \cup \mathcal{S}^{\text{Ts}}$.

Figure 10: Effect of task ordering. Reference (red) is Indep baseline, showing error bars when varying the seed used to initialize the networks. The other baselines are FT-prev with a) the default ordering (yellow), b) with within year shuffling of tasks (purple), and c) with shuffling across the entire stream (green).

Results are showin in Fig. 10. We have found that shuffling the task ordering within a year does not affect the error rate in average, meaning that if we average across several stream variants that differ in the within year shuffling we obtain a similar error rate to the default version of Nevis'22.

Randomizing the order of the tasks over the *entire* stream has instead more dramatic effects. Whenever larger datasets like ImageNet are moved to earlier times, the average error rate is lower. The average error rate over 3 random orderings is 0.253, while the average error rate when shuffling withing years is 0.260 when using FT-prev that accrues knowledge over time. Recall that the standard deviation of the Indep baseline is 0.003 (also displayed on the figure). Overall this suggests that the order of the tasks does affect performance in Nevis'22, and that current baselines struggle to transfer from several smaller datasets to bigger datasets.

5.12 Other Stream Variants

In Tab. 5 we study how tasks in $\mathcal{S}^{\text{Tr}}$ affect performance of FT-d on $\mathcal{S}^{\text{Ts}}$. We picked FT-d since this is a baseline whose performance on $\mathcal{S}^{\text{Ts}}$ is expected to depend on what it has learned on $\mathcal{S}^{\text{Tr}}$. The average test error using the default version of $\mathcal{S}^{\text{Tr}}$ is 0.273. If we remove ImageNet, the average error rate increases by 3%, which is not surprising as the network trained on ImageNet is selected for finetuning by several subsequent tasks, as shown in Fig. 18 of Appendix. Shortening $\mathcal{S}^{\text{Tr}}$ by selecting only tasks belonging to the major domains or larger datasets increases the error rate slightly. FT-d does not seem to leverage well smaller datasets and minor domains. Without improving the transfer ability, this baseline could in fact strike a better trade-off by removing tasks from $\mathcal{S}^{\text{Tr}}$ as shown when removing the smallest datasets (with less than $10,000$ training examples).

Baseline Name	Total Tasks in $\mathcal{S}^{\mathrm{Tr}}$	Avg Error
Full Stream	79	0.276
Full Stream excluding ImageNet	78	0.303
Large Datasets Only	40	0.270
Random 40 Tasks	40	0.282 ± 0.01
Major Domain Datasets Only	43	0.285
Random 43 Tasks	43	0.286 ± 0.01
Remove First 30 Tasks	49	0.295
Remove Last 30 Tasks	49	0.283
Remove Random 30 Tasks	49	0.283 ± 0.009

Table 5: Average test error rate in $\mathcal{S}^{\mathrm{Ts}}$ using FT-d. Each row correspond to a different variant of $\mathcal{S}^{\mathrm{Tr}}$; the first row is the default version of Nevis'22. Baselines of random tasks are run with 5 random seeds and we report the mean and std of average test error rate.

Notice that the failure of FT-d to transfer from several small datasets to a bigger dataset is not a limitation of Nevis'22, but a limitation of current approaches. Nevis'22 detects such deficiency and it enables the discovery of methods that might transfer also in this more difficult (but not so uncommon) condition. It is up to a method to figure out what to leverage when learning on a given task. The data and evaluation protocol provided by Nevis'22 are independent from any particular modeling choice, which includes what and how to transfer from.

The last section of tab. 5 shows what happens when we remove 30 tasks, either at the beginning, at the end or at random from the meta-train part of the stream. The first 30 tasks from the stream contain tasks that many subsequent tasks are finetuned from, including ImageNet, Caltech256, Scene8, etc. Therefore, removing the first 30 tasks from the stream deteriorates performance the most.

5.13 Forward Transfer

An ideal never-ending learner should be able to learn faster by transferring knowledge from the past to the future. In this study we compare two fine-tuning approaches, namely FT-prev and FT-s, in terms of forward transfer. There are 9 tasks that are presented twice in the full stream. For each of these tasks, higher forward transfer should imply faster learning when the same task is presented the second time. Notice that the learner has to figure out task relatedness even on duplicate tasks, as every task (including duplicates) is assigned a unique task id and classification head. To measure forward transfer, we adapt the metric proposed in Wolczyk et al. (2021) by computing the normalized difference between the area under the first learning curve and the area under the second learning curve:

$$\mathrm{FWT} := \frac{\mathrm{AUC}_2 - \mathrm{AUC}_1}{1 - \mathrm{AUC}_1}, \tag{5}$$

where AUC_1 and AUC_2 are the areas under the accuracy curves on the evaluation dataset when the task was presented for the first and the second time, respectively. The resulting

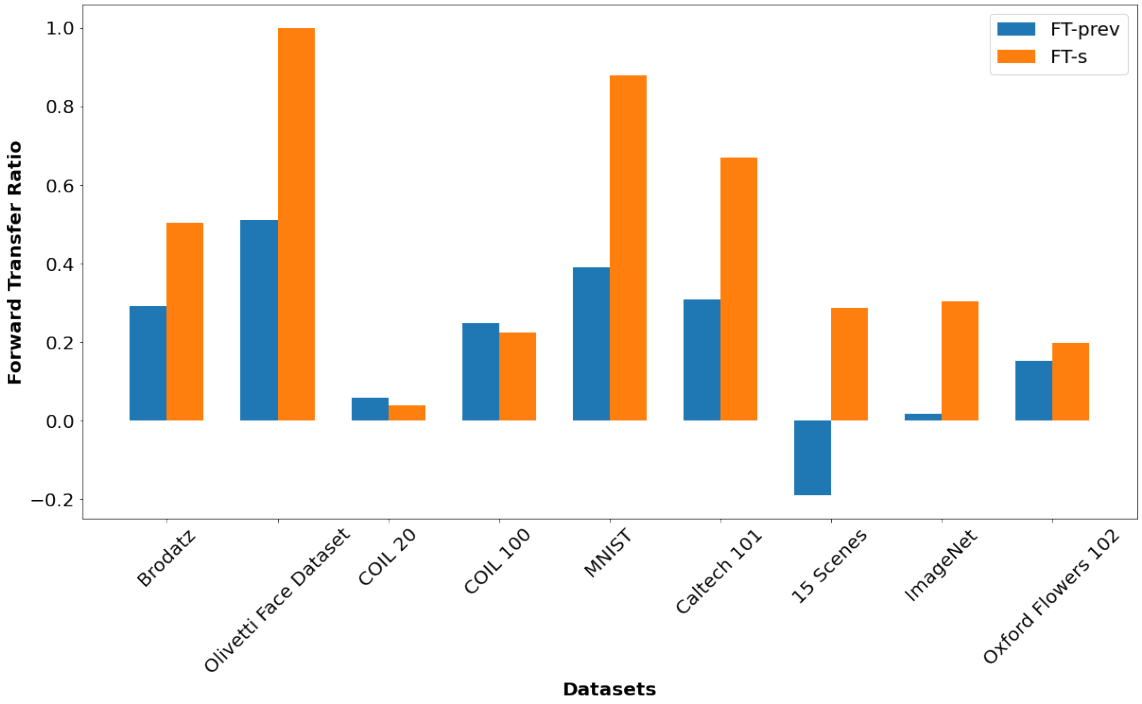


Figure 11: Forward transfer performance of two different fine-tuning strategies.

metric is less or equal to 1, and higher value indicates better forward transfer. Fig. 11 shows that these FT learners do achieve positive forward transfer in average, with FT-s outperforming FT-prev. Averaging the forward transfer measure across the repeated tasks yields **0.199** for **FT-prev** and **0.456** for **FT-s**. In fact, on some tasks like Olivetti and MNIST the forward transfer of FT-s approaches 1.0. Since FT-prev finetunes from a possibly interfering task, on the 15 scenes dataset it obtains an even negative transfer, highlighting again the importance of estimating task relatedness. [7]

5.14 Split-ImageNet

In our last study, we apply the same baselines and training and evaluation protocol to another stream, Split-ImageNet, (Rebuffi et al., 2017b; Wu et al., 2019). Its much smaller variant, Split-MiniImageNet, is one of the most popular large-scale streams used in continual learning research (Shim et al., 2021; Mai et al., 2022). Split-ImageNet is a stream derived from ImageNet, where the original 1000 classes are partitioned into 100 disjoint groups, creating a stream of 100 10-way classification tasks. The goal of this study is to assess whether this benchmark yields similar findings as Nevis'22.

From Fig. 12, it can be seen that the difference between the approaches dramatically reduces as the computation budget increases. Moreover, the ranking of the approaches is rather different. On Split-ImageNet FT-prev performs the best. This is not surprising since tasks in Split-ImageNet are highly related and very homogeneous. This however is an artifact

7. FT-d is omitted from this plot because its behavior in terms of transfer is closely related to FT-s. It offers a way to estimate task relatedness on the fly, and the same observations and conclusions as for FT-s hold.

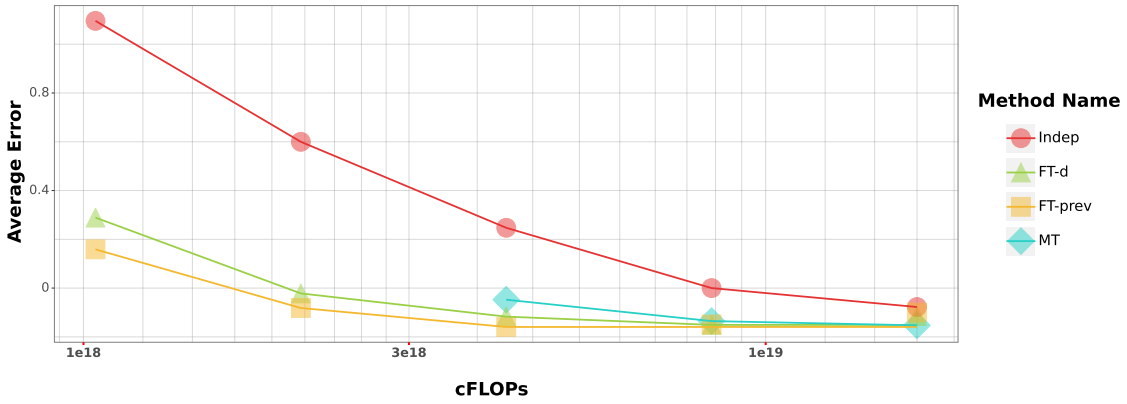


Figure 12: Results on the Split-ImageNet stream with 100 tasks.

of the lack of diversity of the Split-ImageNet stream, and it highlights the usefulness of the proposed Nevis'22 benchmark.

6. Ethical Considerations

A potential concern about the methodology used to build Nevis'22 is the use of relatively old datasets that might suffer even more greatly from issues that our community only recently has started to analyze. For example, face datasets have been found to lack demographic representation across characteristics such as race and gender, leading to disproportionate lower performance on these subjects (Prabhu and Birhane, 2021). Additionally, some machine learning datasets scraped from the internet have been collected without subject consent, prompting to privacy concerns in the collection process (Paullada et al., 2021).

The standards and criteria used to build datasets evolve over time, and therefore, is it sensible to even consider a stream built upon historical datasets which might not meet today's best practices on issues such as representation and consent? We posed ourselves this question and concluded that the proposed training/evaluation protocol offers a sufficient mitigation. In particular, the ultimate evaluation is on the meta-test part of the stream, which only includes tasks extracted from the last three most recent years. Moreover, we plan to update the benchmark on a regular basis, to maintain a fresh meta-test stream. This not only alleviates potential model overfitting, but also it enables Nevis'22 to track community's standards for what datasets one should consider for evaluation purposes.

A related question is around deprecated datasets. To the best of our knowledge, we have removed any deprecated dataset during the construction of Nevis'22, or used the most recent non-deprecated version of a dataset (e.g., most recent version of ImageNet). However, it is possible that at some point in the future a dataset currently in Nevis'22 will be deprecated. In that case, we will take the responsibility to update the stream by removing any instance of such deprecated dataset in the stream.

From a machine learning point of view, deprecation raises a very interesting question of how to remove knowledge of a particular dataset from a never-ending learning system which presumably has accrued knowledge over time, including from that particular deprecated dataset. While we do not have an answer to this question, we believe that Nevis'22 offers an

excellent realistic environment to assess whether methods are capable of such manipulation of learned knowledge.

We also reflected on the potentially harmful downstream use cases related to vision tasks, specifically tasks related to facial recognition. Our objective is to enable the development of a core capability of an AI system as opposed to target a particular application, such as face recognition, In particular, our models are trained in a closed world setting, meaning that a classification model trained on a dataset in Nevis'22 cannot be used to recognize faces of subjects outside those present in the training set, greatly alleviating concerns related to misuse of models trained on this data for surveillance related applications. Given all these considerations we opted for keeping face datasets that satisfied our requirements on deprecation.

By virtue of the methodology used to construct the stream which relies on existing datasets, we acknowledge the above mentioned limitations of Nevis'22. Further progress is certainly required on data collection methods to tackle issues of representation, deprecation, consent, and potential misuse.

However, we overall believe that the net outcome of this research is positive for our research community and society at large, as Nevis'22 encourages the design of more computationally efficient models, that better leverage previous knowledge to learn the next thing more quickly. Given the amount of resources that large-scale models consume, we believe that taking such perspective is very important and it will be even more important in the future as the community further scales up foundation models. While it is still an open question how to effectively learn sequentially while saving compute, the requirement to explicitly measure not just accuracy but also the compute will encourage researchers to strike better trade-offs than it is otherwise possible today.

7. Conclusions and Future Work

In this work, we introduce Nevis'22, a benchmark for evaluating life-long learners on a stream of visual classification tasks. These have been derived by uniformly sampling papers from major computer vision proceedings over the last three decades. Since each task is well understood, the main challenge is learning over time to accrue and transfer knowledge. Only by doing so, can learners become more accurate and efficient over time.

Nevis'22 comes equipped with a rigorous training and evaluation protocol that is designed to prevent overfitting to the evaluation set. In particular learners are asked to go through the tasks of the last three years only once, and without accessing data of future tasks. Moreover, the evaluation consists of not only an assessment of the classical generalization error, but also compute in terms of FLOPs. Had we not controlled for compute, results would have been different and less revealing, as beating method A with method B would be easier once we provide B with more compute than A. Finally, Nevis'22 makes meta-learning a first class citizen, since the assessment of the compute spent while learning *includes* the compute spent while doing hyper-parameter search. Therefore, methods that have more efficient meta-learning algorithms will be favored.

In general, Nevis'22 is not just about a particular stream, but it is also *a process* to build benchmarks. A similar construction method could have been used on other domains, like natural language processing or reinforcement learning, for instance. Nevis'22 is open-sourced

with scripts to recreate the data stream, the training and evaluation framework and code implementing the most classic baselines.

Our initial results obtained by applying standard baseline approaches to Nevis'22 demonstrate the importance of using such a diverse stream. We have found that methods that do transfer perform better than methods that do not, although results vary significantly by domain, image resolution and number of training examples. In particular, we have found that methods that shuffle data to learn generic representations currently strike a worse trade-off between error rate and compute than smarter versions of finetuning. Pre-training approaches perform very well, but they can achieve even superior trade-offs by adapting their representation over time.

Nevis'22 opens up several avenues of future research in never-ending learning. One direction is towards architectures that support variable resolution inputs like the recent Perceiver (Jaegle et al., 2021) and architectures that support efficient inference despite the large number of parameters, like mixture of experts models (Gross et al., 2017). Another direction is on learning algorithms that enable better transfer, model growth over time (Caccia et al., 2022; Gesmundo and Dean, 2022), and parameter sharing across related tasks (Rebuffi et al., 2017a). Another avenue is meta-learning to better answer questions about how to initialize predictors, how to learn more quickly future tasks and how to better shape the search space of architectures, optimizers and learning algorithms. Ultimately, we conjecture that the best method in Nevis'22 will need advances at the intersection of continual learning, meta-learning and AutoML, because it will have to adapt to a non-stationary stream of data, leveraging structure across tasks in order to more efficiently tune hyper-parameters for a new predictor.

In the future, we plan to keep evolving Nevis'22 over time, by moving the current $\mathcal{S}^{\text{Ts}}$ to $\mathcal{S}^{\text{Tr}}$, and forming a new $\mathcal{S}^{\text{Ts}}$ by adding tasks after 2021. This will prevent overfitting and make sure Nevis'22 tracks the community interests and standards. In the near future, we are eager to learn how the community uses Nevis'22 and we want to understand whether there are ways to improve it. Eventually, we plan to extend Nevis'22 towards a multi-task and multi-modal stream, while retaining the same rigorous training and evaluation protocol we have defined in this work.

Acknowledgments

The authors wish to thank Timothy Nguyen and Joaquin Vanschoren for reviewing this work and providing extensive feedback on how to improve its clarity. The authors also thank Skanda Koppula and Iain Barr for their help with training the pre-trained models. This work has been done at DeepMind without any other source of funding.

Appendix A. Individual Contributions

If you wish to contact us, please email us at nevis@deepmind.com. For questions about a specific part of this work, please reach out directly to the relevant authors. Table 6 provides details on each author's contributions.

Authors	**Contributions**
Jörg Bornschein	conceptualization, methodology, codebase development, FT-* baselines, analysis
Alexandre Galashov	codebase development, data pipeline, scripts to fetch data, experiments, debugging, analysis, visualizations
Ross Hemsley	codebase design and development, baselines, data ingestion libraries, open sourcing, write-up
Amal Rannen-Triki	conceptualization, methodology, stream construction, scripts to fetch data, Indep baseline, analysis, write-up
Yutian Chen	scripts to fetch data, BHPO, analysis
Arslan Chaudhry	scripts to fetch data, multitasking baselines, analysis, write-up
Xu Owen He	scripts to fetch data, PT-* baselines, forward transfer ablation, analysis, writing
Arthur Douillard	data ingestion libraries, PyTorch codebase, open sourcing
Massimo Caccia	ensembling baseline, beta testing
Qixuan Feng	SplitImageNet ablation, tensorboard in open-source code
Jiajun Shen	ablation on stream variant (Tab. 5), memory handling in open source code
Sylvestre-Alvise Rebuffi	ViT, discussions
Kitty Stacpoole	program management, coordination
Diego de las Casas	initial design of codebase
Will Hawkins	ethical considerations & review.
Angeliki Lazaridou	discussions
Yee Whye Teh	discussions
Andrei A. Rusu	conceptualization, methodology, formal analyses, writing - review & editing.
Razvan Pascanu	conceptualization, analyzing and discussing results, writing
Marc'Aurelio Ranzato	conceptualization, stream construction, scripts to fetch data, experiments with FT-*, analysis, writing, team coordination and planning

Table 6: Authors contributions.

Appendix B. High level description of main results and methods used

In this section we give an overview of results and methods presented in all the Appendix sections. For more detailed discussions, please refer to each of the appendix sections.

In Section C, we analyze the behavior of different methods under lower computational budget than in the main paper (see Section 5.6). The lower compute budget is achieved by using cheaper arhictectures (ResNet-18 and ResNet-34) as well as using fewer trials in hyperparameters search and fewer updates. Overall, we show that the same conclusion as presented in Section 5.6, hold.

In Section D, we present results showing impact of the architecture choice. We consider different backbone architectures, notably: VGG, ResNet34, ResNet50, Vision Transformers (ViT), in particular, ViT-B8. On top of that, for ResNet-34 architecture we provide study of the impact of the input image resolultion, as well as the number of channels in the first residual block. There is a simple compute-performance trade-off for choosing these values, i.e. - larger resolutions lead to better performance and lower compute, and vice-versa.

In Section J, we explain the method of calculating the transfer matrix which is required for FT-d (see Section 5.6) method as well as to understand the task structure, see Figure 3.

In Section E, we present additional baseline results. In Section E.1, we present ablations of different sequential finetuning baselines, i.e., FT-d, FT-s, FT-prev, as well as the impact of using pretrained models (PT-ext). In Figure 18 and Figure 19, we study the qualitative behaviour of FT-d finetuning method without (FT-d) and with (PT-ext + FT-d) pretraining. In Section E.2, we additionally study the impact of different pre-training and finetuning strategies.

In Section F, we add quantitative results showing the impact of task orders in the stream. We found that the overall performance is very sensitive on the position of ImageNet dataset in the stream.

In Section G, we study the performance of multi-task finetuning strategy. We show that it is very important how the network is initialised when it starts to train on each new task in the stream. Moreover, we found that picking all previous task versus a subset did not lead to a very different performance.

In Section H, we provide more detailed analysis of Bayesian Hyper-Parameters Optimization (BHPO) method. In particular, we show that the benefit of BHPO increases with higher dimensionality of the search space. In low-dimensional (2-D) search space, BHPO performs similarly to Random Search.

In Section I, we present additional results showing that simple ensembling strategy added on top of the reference model could improve results. This, however, comes at the expense of increased inference cost.

In Section K, we explain the reasonning behind choosing cumulative-FLOPS (cFLOPS) as compute metric as well as alternatives which we considered.

Appendix C. Baselines at a Lower Computational Budget

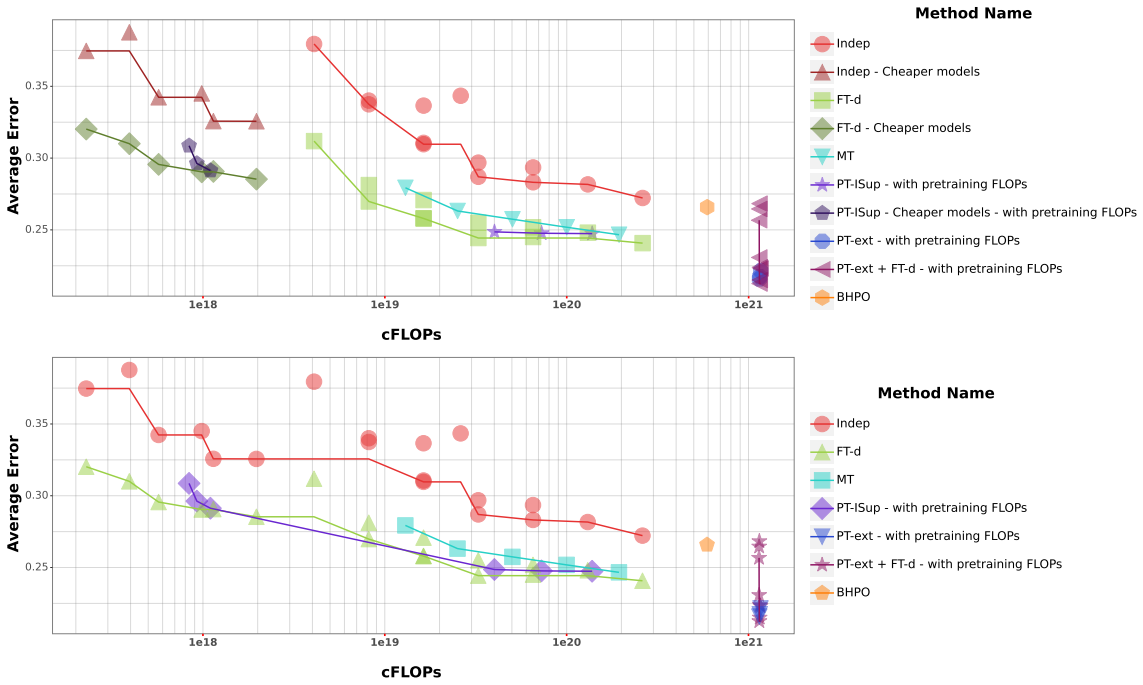


Figure 13: Top: Pareto fronts when considering standard ResNet18 and ResNet34 trained with fewer trials and number of updates (cheaper models). The experiment represented by the third marker of FT-d on the far left ran for 28 hours on the full stream when distributing the hyper-parameter search of the predictor over four devices. On a single worker, the same experiment takes about four days. Bottom: The same pareto front as above, but using a single curve for FT-d, PT-ISUP and Indep.

In the experiment of Fig. 13 we explore another region of the pareto front: the extremely frugal learners that require between 10^{17} and 10^{18} to run on the full stream. This amounts to about 4 days on a single GPU with 16GB of RAM. Unlike what we presented in the main paper where we used a ResNet34 adapted to low-resolution images (with many more channels and without as much spatial sub-sampling), here the network is a default ResNet18 and ResNet34 architecture, which have been designed for higher resolution images. These architectures are cheaper because a) they have at most the same number of blocks, but b) they have many fewer channels. We further reduce compute by fixing the label smoothing parameter to the mid value of our search range, 0.15, and by searching only over four values of learning rate, namely {1e-4, 1e-3, 1e-2, 1e-1}.

We can see that these cheaper architectures and hyper-parameter strategy extend the Pareto front at lower budgets. This suggests that another possible avenue of future work is the design of more efficient architectures and hyper-parameter search methods. Importantly, this might provide an easy entry point to Nevis'22 for researchers who have limited computational resources at their disposal.

Appendix D. Architectures

In this section, we study the impact of the architecture choice. We focus on the independent baseline used as reference in all our experiments, and we conduct two experiments. In the first, we use different architecture variants: VGG, ResNet34, ResNet50 and Vision Transformers (ViT) Dosovitskiy et al. (2020). In this manuscript we use a ViT-B8 with the modifications proposed by He et al. (2022) where the classifier is applied after global average pooling over the vision tokens. As all the experiments are conducted using the same resources, the bigger architectures (ResNet50 and ViT) are run with a smaller maximum batch size (see Eq. (4)), and similar search spaces for the number of updates and learning rate. It is therefore expected to see these architectures underperforming, as we observe in Fig. 14.

The second experiment focuses on ResNet34, chosen as default architecture, and varies the number of channels in the different residual blocks. The results of this experiment complete Fig. 9, providing another axis to trade off error rate versus compute. The full results are shown in Fig. 15. In the legend, rez-*r*x*r* indicates the input resolution, ch-*c* corresponds to the number of channels in the first residual block (the number of channels in the remaining blocks are consecutively doubled), and bsz-*b* shows the maximum batch size. We observe that reducing the number of channels saves a significant compute budget, without a significant loss in performance.

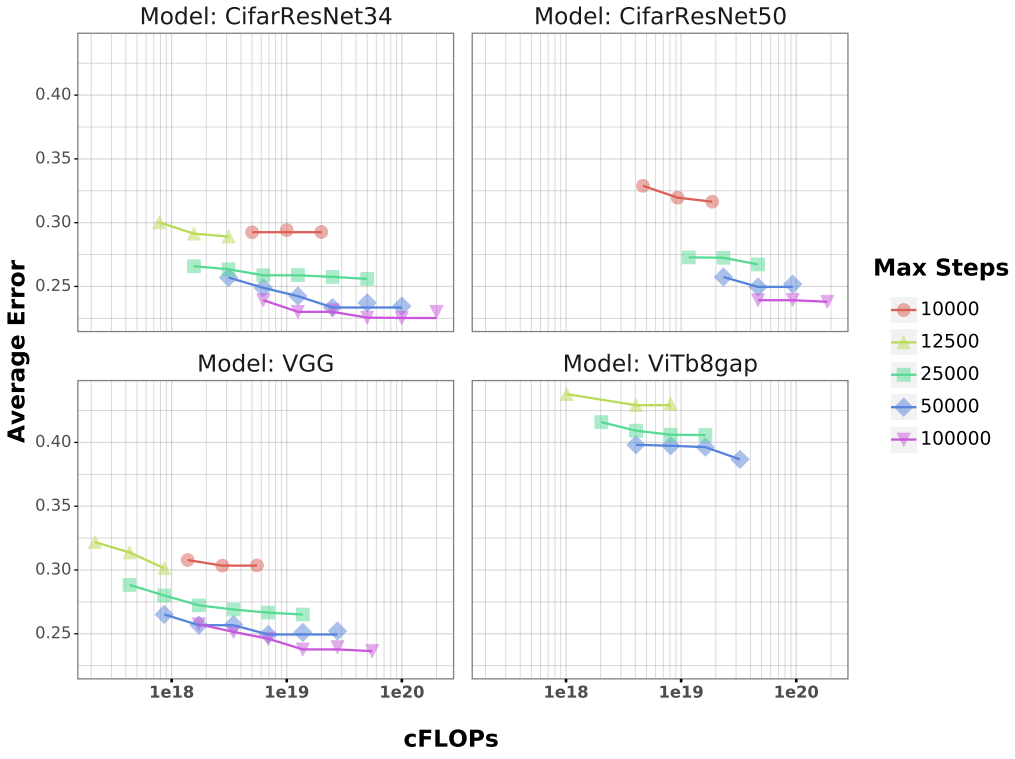


Figure 14: Independent learners with different architecture variants.

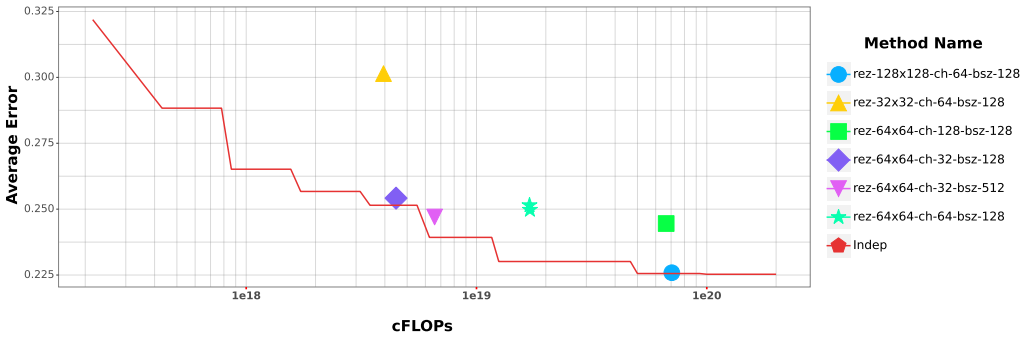


Figure 15: Performance of Indep when varying the input image resolution and the network width. In the legend, ch-c corresponds to the number of channels in the first residual block. The number of channels in the remaining blocks are consecutively doubled.

Appendix E. Additional Baseline Results

In this section, we report additional results on the Nevis'22 stream. We namely show the results of the finetuning and pretraining families, and complete the experiments reported in Sec. 5.6.

E.1 Finetuning

This section focuses on the Finetuning family. In Fig. 16, we show the Pareto fronts of FT-d, FT-prev and FT-s. Fig. 17 reports the regret plot of the same methods, i.e. the cumulative error over time relative to Indep, picking hyper-parameters such that all methods use roughly the same amount of compute (same as in Fig. 5). These results demonstrate the importance of the choice of what to finetune from, as FT-prev is significantly worse than FT-s and FT-d, and FT-s is worse than FT-d suggesting that it is important to make an accurate (up-to-date) estimate of task relatedness. The regret plot further shows that the tested finetuning strategies fail to increase transfer over time (linear slope over the full stream) and to accumulate knowledge over the last 9 tasks of the sequence (horizontal curves).

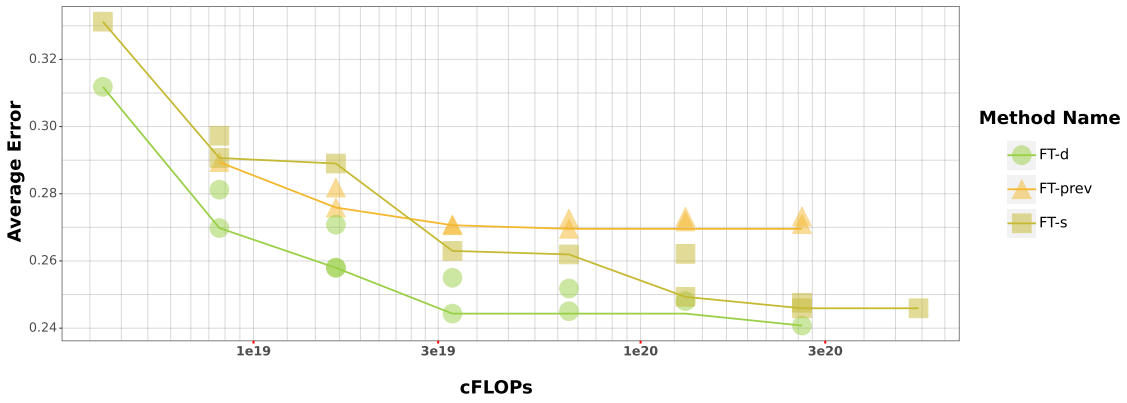


Figure 16: Finetuning baselines: Pareto fronts

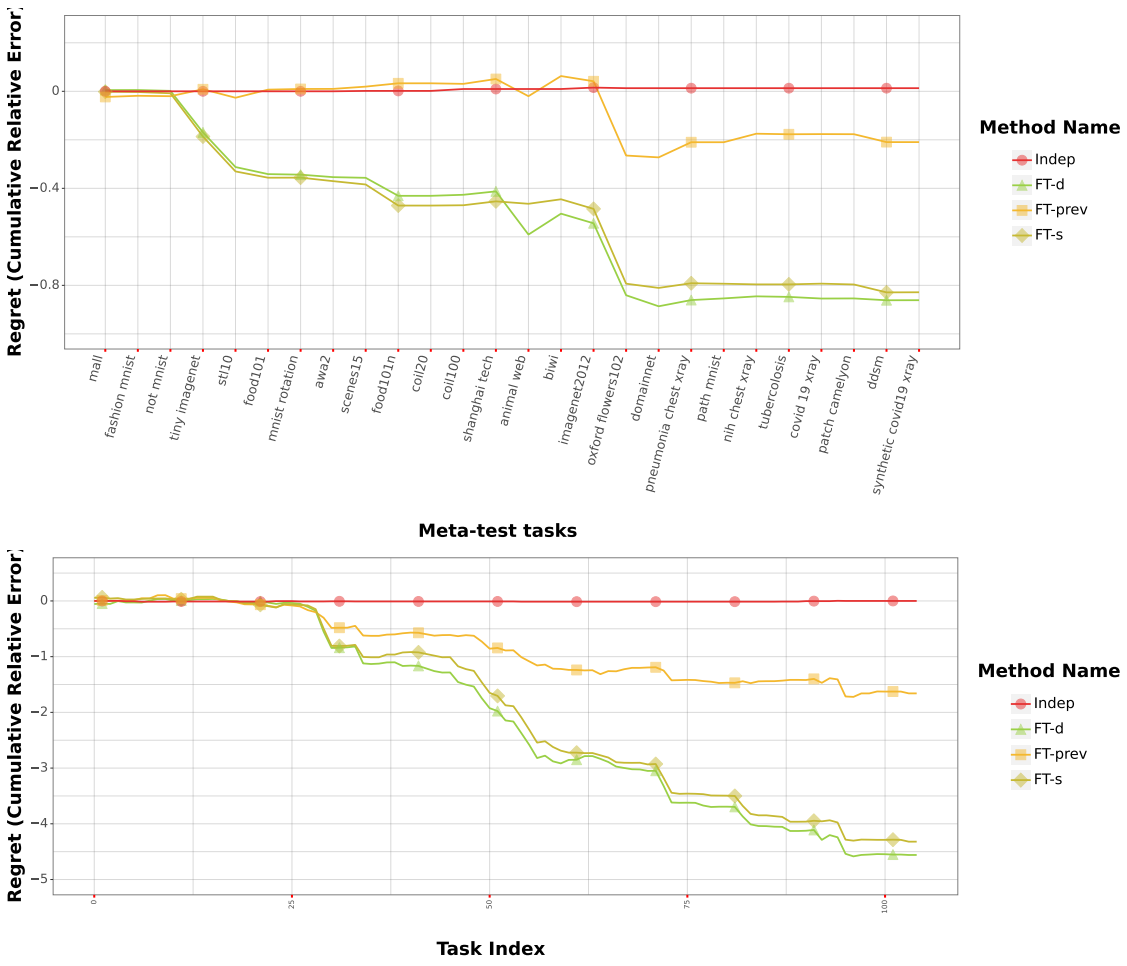


Figure 17: Finetuning baselines - Regret plots: Cumulative error rate relative to Indep. on $\mathcal{S}^{\text{Ts}}$ (top) and on the full stream (bottom)

E.1.1 Analysis of FT-d

In Fig. 18 we report an example of the finetuning sequence learned by FT-d (this is the version which was trained on every task with $50,000$ steps using 16 trials of hyper-parameter search). We notice that there are several hubs from which several other models are finetuned from. The biggest hub is ImageNet, followed by Caltech256, MNIST, Caltech101, Scene8, etc. In the graph tasks are mostly organized by visual similarity and domain. For instance, we see two clusters of OCR tasks in red, cluster of medical images in cyan, and a large cluster of generic object recognition tasks in yellow. Finally and perhaps most surprisingly, we observe fairly long chains of finetuning models. It seems that finetuning even more than ten times can produce high performing models.

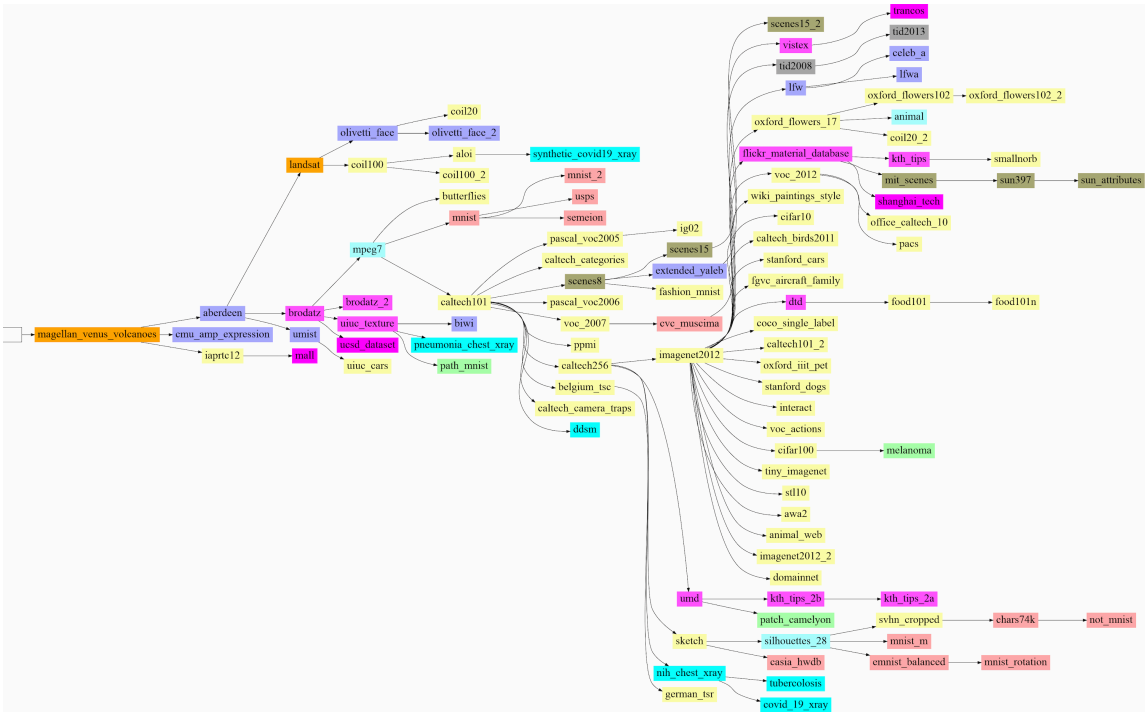


Figure 18: Graph showing the inner working of FT-d. Each box correspond to a task, the color represents the domain. An arrow connecting task i to task j indicates that task i was selected as initialization for the network trained on task j.

E.1.2 Analysis of PT+FT

In Fig. 19 we report an example of the finetuning sequence learned by PT+FT, i.e., FT-d starting from the model pretrained on external data (PT-ext). We report the version which is reported in the regret plots in Figures 5 and 21, which corresponds to the best compute-error trade-off in Figures 4 and 20.

Compared to the finetuning sequence learned by FT-d when learning from scratch (see Fig. 18), we observe that while chains are shorter, not all tasks choose to be initialized from the original pre-trained model. ImageNet is still a strong hub, but there are many chains of length two or three that cluster datasets by domain and visual similarity.

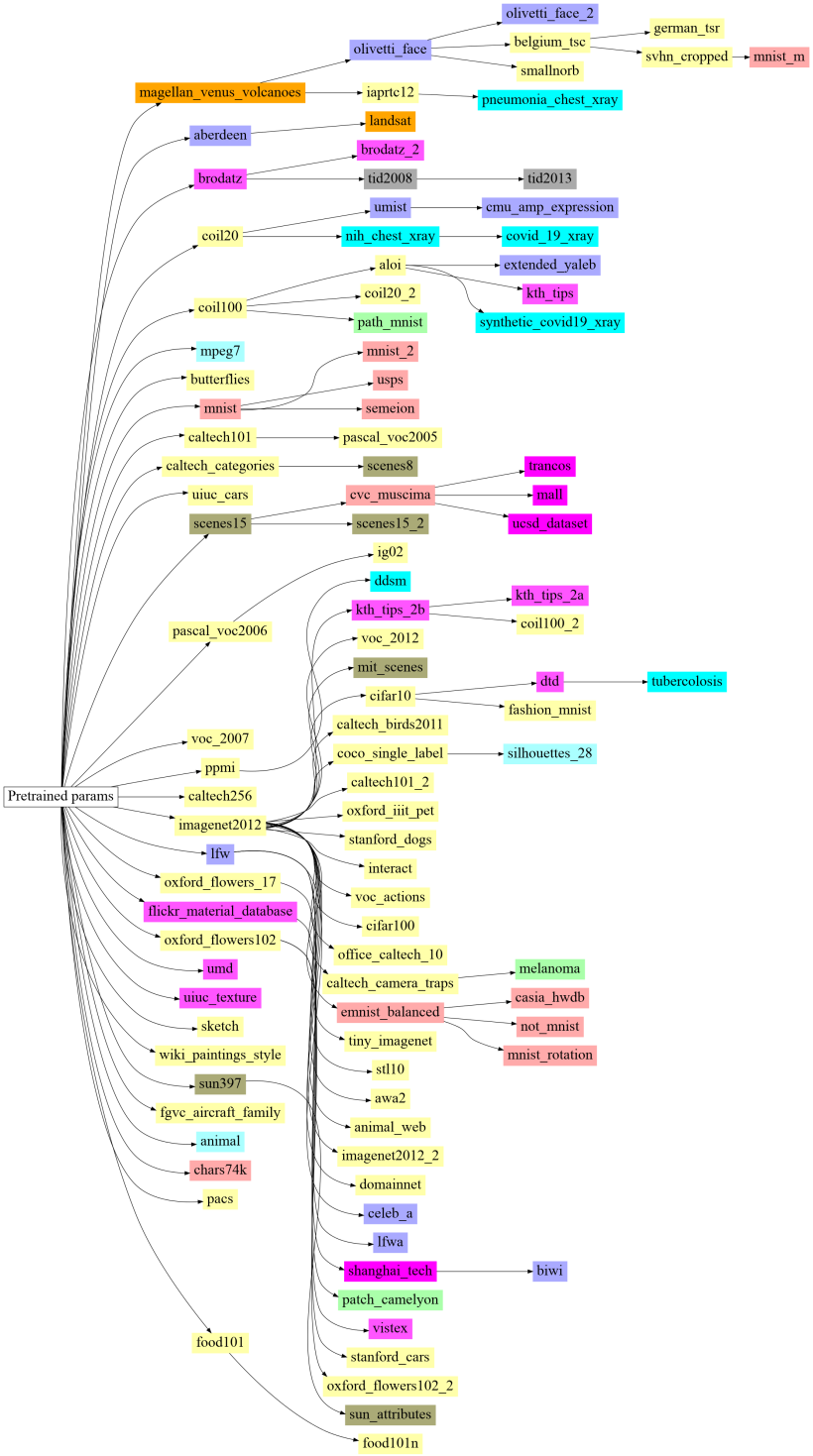


Figure 19: Graph showing the inner working of PT+FT (i.e., FT-d starting from a pretrained model). Each box correspond to a task, the color represents the domain. An arrow connecting task i to task j indicates that task i was selected as initialization for the network trained on task j. Compare this graph to the one of FT-d (starting from scratch) in Fig. 18.

E.2 Pretraining

This section focuses on the Pretraining family. Below, we report results using models pretrained on ImageNet using full supervision (PT-ISup), on the meta-train part of the stream (PT-MT), and on ALIGN and Stock using CLIP (PT-ext). Note that this last variant uses not only a much larger (external) dataset, but also a substantially more powerful architecture.

In Fig. 20, we show the Pareto fronts of these methods, with (bottom) and without (top) the computational cost of pretraining. As expected, we observe that PT-ext leads to a significantly higher performance. However, when the pretraining cost is taken into account, the trade-off between performance and compute is less impressive. We also observe that training on the whole training part of the stream (which includes ImageNet) performs worse than pretraining on ImageNet only.

In Fig. 21, we report the regret plots of the pretraining methods on the last 3 years (top) and the full stream (bottom) with respect to the Indep baseline. As none of these methods accumulate knowledge, it is not surprising to observe that their transfer does not improve over time. However, it is more surprising to observe that they are on a par with Indep for the last tasks (horizontal curves). This shows that even the pretrained model on the largest dataset (PT-ext) fails to transfer well to the medical datasets that appear towards the end of the stream.

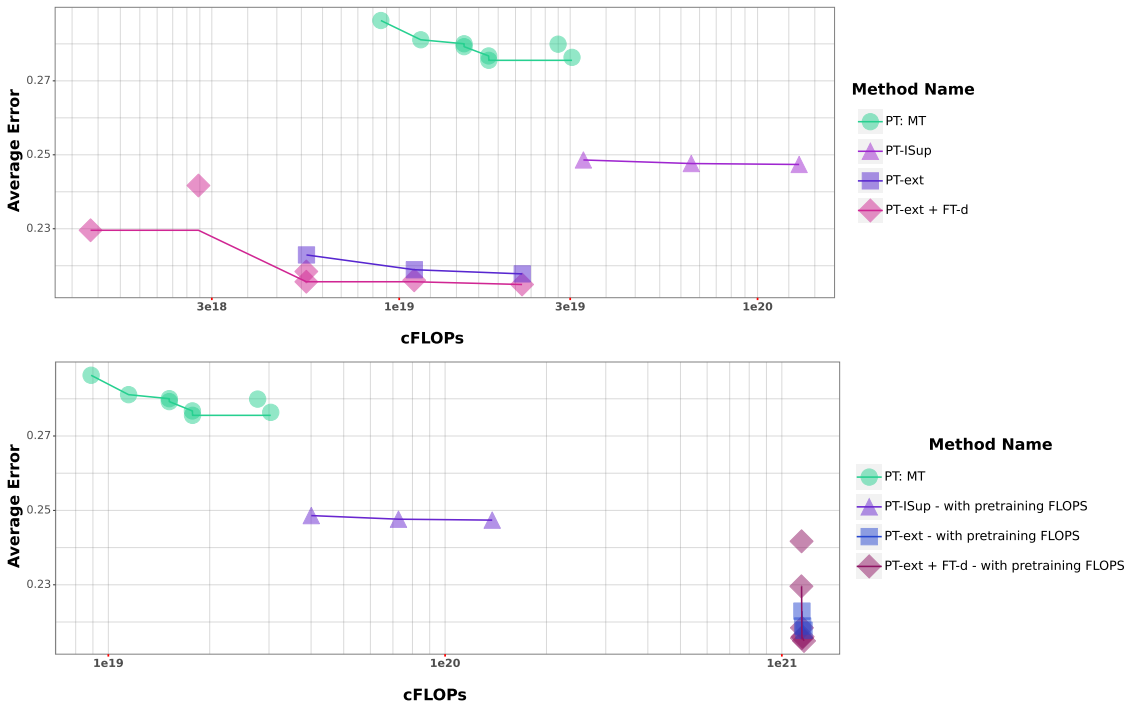


Figure 20: Pretraining baselines: Pareto fronts before (top) and after (bottom) accounting for the flops using during pretraining.

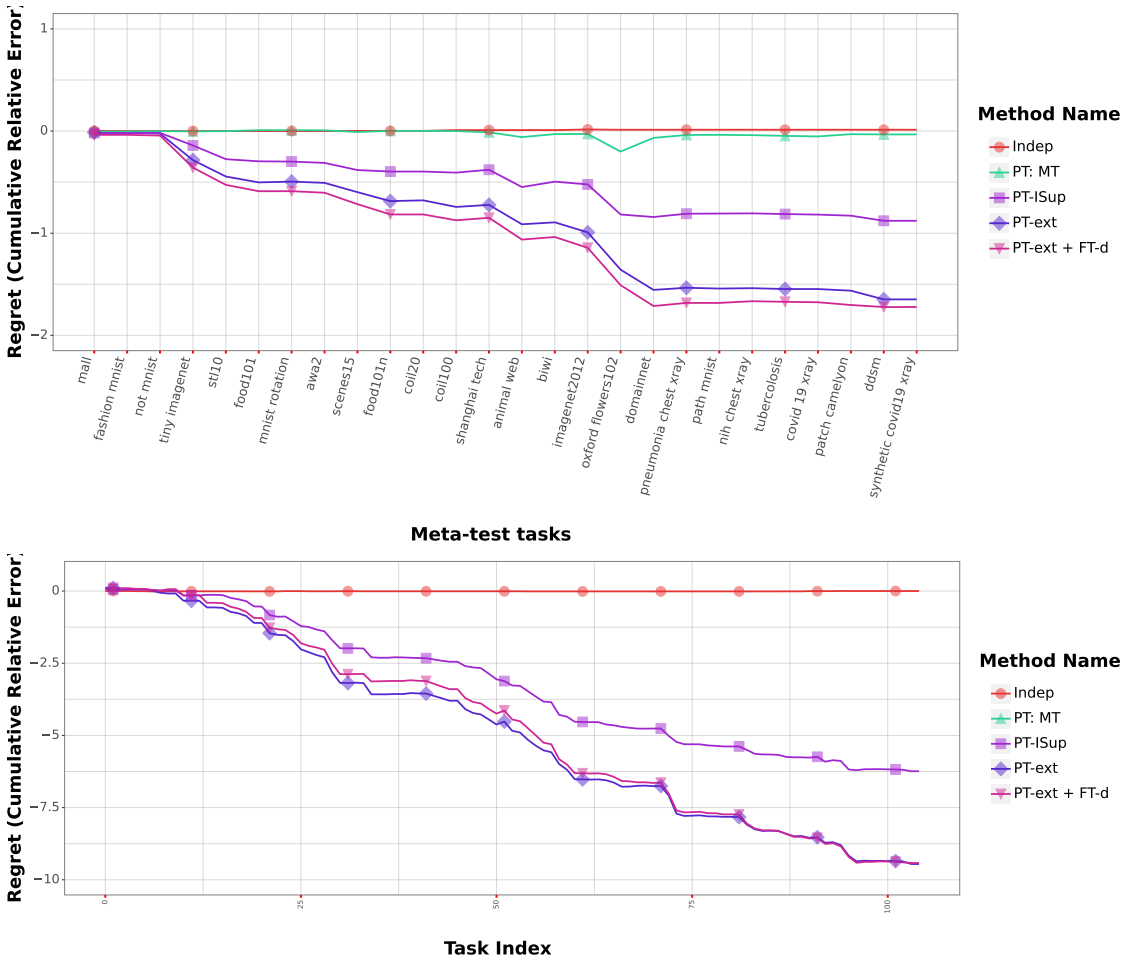


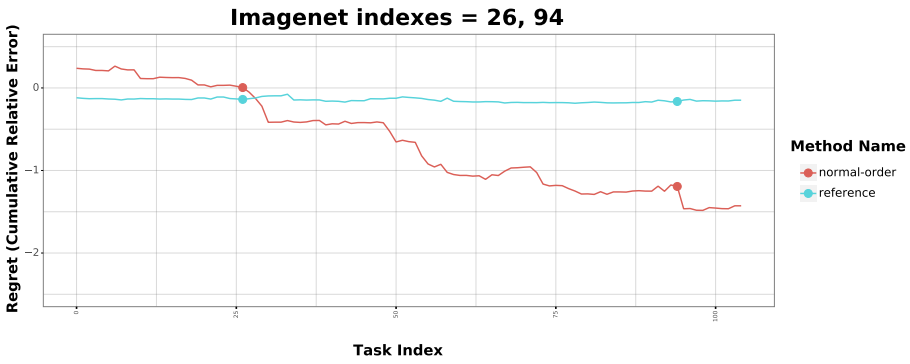
Figure 21: Regret plots: Pretraining baselines

Appendix F. Task Ordering

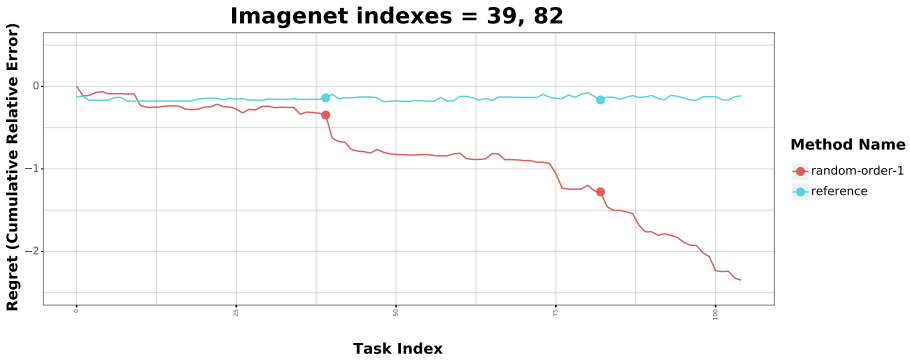
This section completes the results reported in Fig. 10. In Fig. 22, we show the regret plots with respect to the Indep baseline for different random stream orderings. corresponding to the "Random Order (All stream)" experiments in Fig. 10. The positions of ImageNet in the stream are displayed on top of each of the panels, and highlighted on the curves. These results show how the position of this dataset, which dominates the stream in terms of size and complexity, influences the performance. The experiment where ImageNet appears at the beginning of the stream corresponds to the best performance reported in Fig. 10.

Appendix G. Multitask Ablation

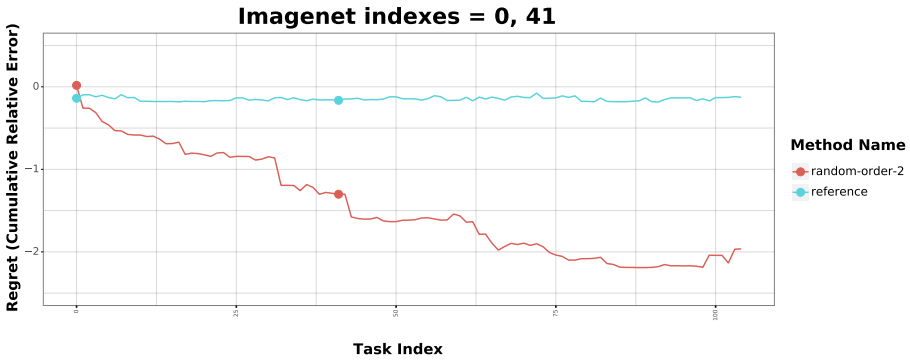
In Fig. 23 we show how various hyper-parameters choices affect the performance of MT. In particular, we observe that picking all tasks versus sampling a subset of them does not yield a better trade-off, but merely extends the Pareto front towards higher compute regimes. We also observe a very large gap due to how the multitask network is initialized. Despite the



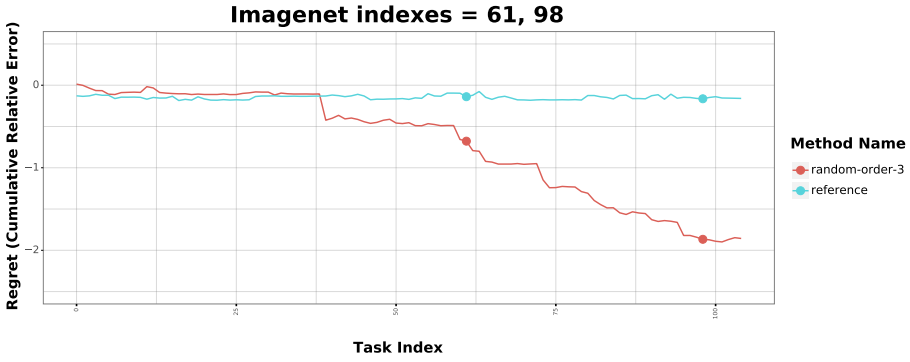
(a) Normal order



(b) Random order (1)



(c) Random order (2)



(d) Random order (3)

Figure 22: Regret plots for different stream orderings.

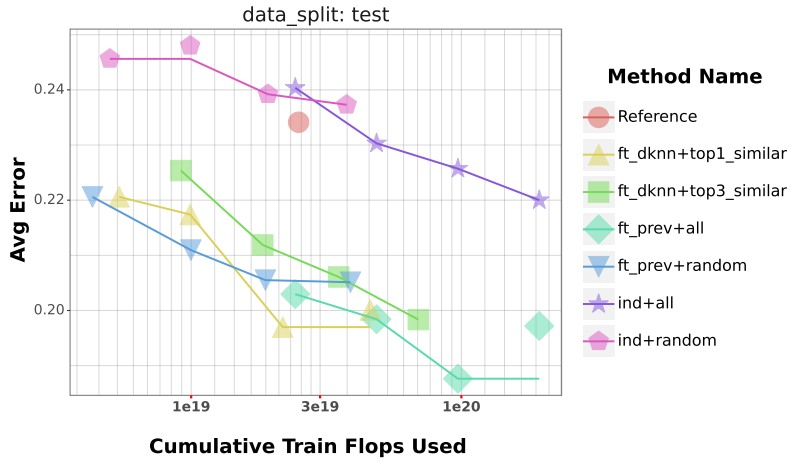


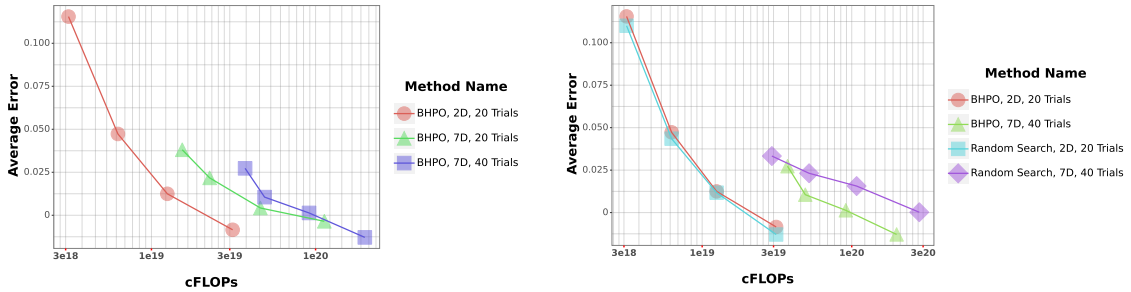
Figure 23: Pareto front of meta-learners using different multitasking strategies on SHORT stream. `ft_dknn+top1_similar` refer to multitask baseline where the meta-learner initializes the predictor of a new task using most similar previous task using dynamic KNN transfer matrix, and use only $k = 1$ auxiliary task. Similarly, `ft_dknn+top3_similar` uses $k = 3$ auxiliary tasks. `ft_prev+all` and `ft_prev+random` refer to multitask baselines where the predictor of a new task is initialized form most recent previous task's parameters and meta-learner either uses "all" the previous tasks as auxiliary tasks, or randomly picks one of the previous tasks as an auxiliary task during an SGD update. Finally, `ind+all` and `ind+random` refer to multitask baselines where the parameter are initialized randomly and either all or one of the previous tasks are used as auxiliary tasks.

multitask learning objective, it is very beneficial to initialize the network, even using the parameters of the most recent previous task. While there is no best initialization across all the compute budgets, i.e., no initialization dominates all the others, we have found FT-d strategy to work better in the intermediate compute budgets. Further, in that setting, it is empirically best to co-train with the task the network is finetuned from (top-1), as opposed to co-train with other tasks as well (top-3). Hence, we use FT-d (top-1) as the MT strategy while reporting the metrics in the main paper.

On architectures that use BatchNorm Ioffe and Szegedy (2015), we found it to be critical for performance to not update the BatchNorm statistics, running means and variances, with the batches of auxiliary tasks. Further, using a small fixed batch size of 64 for all the auxiliary tasks resulted in a compute-efficient learner without hurting the performance as compared to a learner that uses large variable batch sizes for the auxiliary tasks (see eq. 4).

Appendix H. BHPO Ablation

In this section we study the impact of the choice of the search space and optimization algorithm (BHPO versus random search). Using Indep as a case study, we compare two search spaces, a small search space with only two hyper-parameters (learning rate and label smoothing as in the default setting) and a larger space with 7 hyper-parameters which include also how data is augmented, the network architecture, etc. We also vary the number of



(a) BHPO with different search spaces and trials

(b) BHPO versus Random Search.

Figure 24: Pareto fronts varying the search space and search algorithm. Points along each line corresponds to 5K, 10K, 20K and 50K training steps per trial.

training steps per trial (5K, 10K, 20K, 50K) and the number of trials (20, 40). As shown in Fig. 24a a large search space allows one to find a smaller error rate given the same training steps. This is mostly evident with a small number of training steps when the training has not converged yet. With a larger number of steps per training, the advantage diminishes. Also, it becomes harder the find the optimal hyper-parameter setting in a larges space given the same number of trials. Moreover, as the large search space includes optimizing the batch size, the computation cost is much higher. In Fig. 24b we compare random search with the more sophisticated BHPO method. The latter finds a better setting in most cases in the large search space where the optimization is more difficult, but there is no clear difference in the easier 2-parameter space.

Appendix I. Ensembling

In this section, we start by observing that during random hyper-parameter search we perform N trials in parallel, one for each particular configuration of hyper-parameters. Since a hyper-parameter search often yields a set of accurate and diverse models, it seems wasteful to retain only the model that performed the best on the validation set. Instead, we can create an ensemble from these already trained models Dietterich (2000).

Fig. 25 shows that we can significantly lower the error rate by ensembling, or for the same error rate we can drastically reduce the compute. For instance, by ensembling networks trained on 4 trials only, we can attain lower error rate at half of the training compute of a (single component) baseline trained with 8 trials.

In our experiments we have found that the best ensembling approach uses a weighted sum of probability distributions. The weights are the output of a softmax with temperature equal to 0.1 using as input the top-1 accuracy obtained during cross-validation. This has the effect of weighting more the top performing models.

Notice however that while ensembling reduces the training compute at a given level of error rate, it increases linearly the cost at inference time.

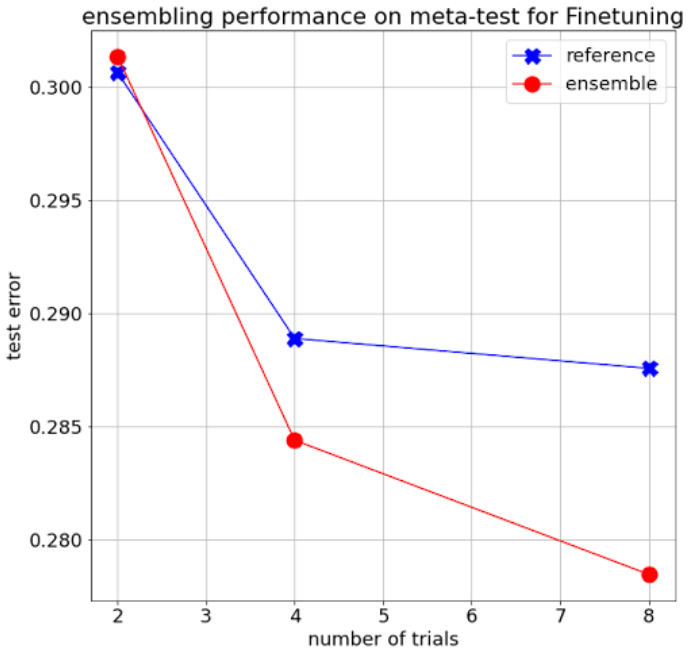


Figure 25: Error rate of ensembling as a function of the number of components. The reference is FT-d using only the best model found during random hyper-parameter search.

Appendix J. Experimental protocol for transfer matrix computation

Figure 3 (right) shows the transfer matrix for the tasks in the Nevis'22 stream. Each cell (i, j) displays the test set accuracy for task i after finetuning on task i after training on task j relative to independently training on task i alone. To compute the transfer matrix we first pick the best training run from the standard hyperparameter-sweep for the *Independent* learner on each task. These runs form the reference accuracy for transfer to task j, and provide the starting point for finetuning. For finetuning we take each such run i and finetune it on all tasks j, $\forall\, j > i$ with the standard hyperparameter sweep for the *finetuning* learner. In total we thus performed more than 90,000 training runs to compute the transfer matrix ($106 + \frac{106 \cdot 105}{2}$ learner runs with 16 random hyperparameters each).

Appendix K. Measuring Compute

For the Nevis'22 evaluation, we had the key goal of enabling a fair comparison between approaches that *efficiently* achieve strong results - such as via knowledge re-use or carefully applied meta-learning strategies - versus methods that always re-train from scratch over all data, and perform large hyper parameter sweeps to obtain good results. We explicitly wanted to choose a metric that would favor efficient learners that can scale gracefully as the data and the number of tasks increase.

Some options that we considered include 1) Counting the total number of training examples used (including multiplicity) 2) counting the total number of optimization steps 3) Estimating the time elapsed during training. Ultimately we found problems with all of these

approaches (and other similar approaches). A common issue is that we felt the measurement should not limit in any way the solution space available to those implementing learners. Counting optimization steps may disadvantage learners that take many very small steps, counting the number of examples accessed has the the problem that, in-reality, accessing data that already exists is a relatively inexpensive operation - and it's easy to imagine that this could penalize learners that make use of replay buffers or caches that ultimately increase learner efficiency.

Counting floating point operations also suffers from challenges - notably since most hardware accelerator devices have a fixed maximum *throughput* of floating point operations per second. Achieving this maximum throughput is key to training learners quickly, and making best use of available resources. Learners that use very sparse compute sparingly may ultimately take a lot longer (in wall clock time) to train than learners that can efficiently use the dense compute available in modern accelerators (such as dense matrix multiplications). Extending this further, learners that can efficiently be trained in parallel can require far less wall-clock time to train compared to learners that are inherently sequential. Many real-world applications would prefer learners that make use of large amounts of compute if they are highly parallelizable. Furthermore, the actual number of floating point operations performed is dependent on the underlying hardware platform (for example due to padding vectorized computations), and the levels of optimization performed on the linear algebra primitives themselves.

When considering our approach, we ultimately kept in mind the following core constraints, 1) the approach should not penalize or limit any particular learner implementation, 2) the approach should reasonably approximate cost of training the learner in a way that maps to reality, 3) storage and memory are typically cheaper than overall numerical compute, and so can be ignored without affecting the ordering of results, 4) it should be practically feasible for users of the benchmark to compute comparable values for the resources used, independently of the hardware they have available. In these respects, we feel that the cumulative floating point operations used offers a fair compromise. We obtained the FLOP counts reported in this study with the *cost analysis* API provided by the JAX deep learning framework when targeting execution on a CPU. The counts therefore take attained compiler optimizations into account, and include only minimal overhead due to padding for wide SIMD hardware. We confirmed that the reported counts closely match manually estimated counts for various model architectures and image sizes.

Appendix L. List of Datasets in Nevis'22

The following tables list all the datasets included in our benchmark, in the order of use. For each dataset, we indicate the year in which it appeared in our sampling procedure, the sampled paper that uses it, the task type(classification: C or multilabel classification: M) , the image domain, the number of samples in the dataset (size) and the average input resolution.

Year	Dataset Name	Sampled Paper	Type	Domain	Size	Avg. res.
1992	Magellan Venus Volcanoes Ma2,Dua and Graff (2017)	Lazebnik et al. (2006a)	C	satellite	102	(1024, 1024)
1992	Aberdeen face database. Ab1	Craw and Cameron (1992)	C	face	468	(519, 417)
1998	LandSat UCI repo La5,Dua and Graff (2017)	Jones (1998)	C	satellite	3764	(3, 3)
1998	Brodatz Brodatz (1966),Br3	Fountain et al. (1998)	C	texture	672	(213, 213)
1998	Olivetti Face Dataset Samaria and Harter (1994),Ol6	Hall et al. (1998)	C	face	288	(80, 70)
2000	COIL 20 Nene et al. (1996),CO8	Matas et al. (2000)	C	object	973	(128, 128)
2001	COIL 100 Nayar (1996),CO10	Gibson and Harvey (2001)	C	object	6120	(128, 128)
2001	MPEG-7 MP12	Gibson and Harvey (2001)	C	shape	943	(341, 388)
2004	Butterfly dataset Lazebnik et al. (2004),Bu13	Lazebnik et al. (2004)	C	object	460	(335, 431)
2004	MNIST LeCun et al. (1998a),MN14	Brown et al. (2004)	C	ocr	51000	(28, 28)
2005	Caltech 101 Li et al. (2022),Ca16	Bart and Ullman (2005)	C	object	2601	(244, 301)
2005	UMIST Graham and Allinson (1998),UM19	Kong et al. (2005)	C	face	686	(217, 199)
2005	CMU AMP expression CM18	Kong et al. (2005)	C	face	650	(64, 64)
2006	Pascal 2005 Pa23	Hegerath et al. (2006)	C	object	881	(431, 553)
2006	Caltech Categories Fergus et al. (2003),Ca22	Gill and Levine (2006),Hegerath et al. (2006)	C	object	996	(341, 514)
2006	UIUC cars Agarwal et al. (2004),UI24	Gill and Levine (2006)	C	object	823	(50, 112)
2006	ALOI Geusebroek et al. (2005),AL20	Geusebroek (2006)	C	object	71494	(144, 192)
2007	8 Scene Dataset Torralba et al. (2003)	Wang and Gong (2007)	C	scene	1811	(256, 256)
2008	15 Scenes Lazebnik et al. (2006b),Sc70	Cai et al. (2008)	C	scene	3021	(244, 273)
2009	Pascal 2006 Pa83	Weyand et al. (2009)	C	object	2211	(420, 525)
2009	Extended YaleB Ex82	Everingham et al. (2006)	C	object	10688	(480, 640)
2010	Pascal 2007 Pa85	Su et al. (2010)	M	object	2501	(382, 471)
2010	Graz-02 Opelt et al. (2006),Gr84,Marszalek and Schmid (2007)	Zhang and Chen (2010)	C	object	747	(497, 622)
2010	Olivetti Face Dataset Ol6	Chen (2010)	C	face	288	(80, 70)

Year	Dataset Name	Sampled Paper	Type	Domain	Size	Avg. res.
2010	PPMI Yao and Fei-Fei (2010),PP86	Delaitre et al. (2010)	C	object	2023	(258, 258)
2011	Caltech 256 Griffin et al. (2022),Ca88	Kim et al. (2011)	C	object	20696	(325, 371)
2011	ImageNet Deng et al. (2009a),Im90	Kim et al. (2011)	C	object	1281167	(406, 473)
2011	LFW Huang et al. (2007),LF92	Meina Kan and Chen (2011)	C	face	11248	(250, 250)
2011	Oxford Flowers Nilsback and Zisserman (2006),Ox93	Yang and Yang (2011)	C	object	680	(555, 583)
2011	Flicker Material Dataset Sharan et al. (2014),Fl89	Diane Hu and Ren (2011)	C	texture	676	(384, 512)
2011	Oxford Flowers 102 Nilsback and Zisserman (2008),Ox94	Muhammad Awais and Kittler (2011)	C	object	1020	(534, 630)
2011	Belgium Traffic Sign Dataset Timofte et al. (2009),Be87,Mathias et al. (2013),Timofte et al. (2014)	Timofte and Van Gool (2011)	C	object	3887	(118, 105)
2011	German Traffic Sign Recognition Benchmark Stallkamp et al. (2012),Ge53	Timofte and Van Gool (2011)	C	object	33392	(50, 50)
2011	Brodatz Br3	Dahl and Larsen (2011)	C	texture	672	(213, 213)
2011	VisTex Vi54	Dahl and Larsen (2011)	C	texture	125	(512, 512)
2012	UMD Xu et al. (2006),UM59,Xu et al. (2009b),Xu et al. (2009a),Xu et al. (2012)	Timofte and Van Gool (2012)	C	texture	676	(960, 1280)
2012	KTH-TIPS KT56	Timofte and Van Gool (2012)	C	texture	554	(196, 200)
2012	UIUC texture Lazebnik et al. (2005),UI58	Timofte and Van Gool (2012)	C	texture	676	(480, 640)
2012	KTH-TIPS2-b KT57	Timofte and Van Gool (2012)	C	texture	3191	(199, 198)
2012	CVC-MUSCIMA Fornés et al. (2012),CV55	Timofte and Van Gool (2012)	C	ocr	419	(1848, 3465)
2013	IAPRTC-12 Escalante et al. (2010),IA60	Verma and Jawahar (2013)	M	object	10830	(395, 439)
2013	sketch dataset Eitz et al. (2012),sk63	Li et al. (2013)	C	object	13536	(1111, 1111)
2013	KTH-TIPS2-a KT61	Qi et al. (2013)	C	texture	3097	(200, 198)
2013	Pascal 2012 Pa64	Chatfield et al. (2014)	C	object	5717	(386, 470)
2013	NORB LeCun et al. (2004),NO62	Wu et al. (2013)	C	object	20655	(96, 96)
2014	Wikipaintings Tan et al. (2019),Wi65	Karayev et al. (2014)	C	object	48576	(223, 221)
2015	MNIST LeCun et al. (1998a),MN14	Srinivas and Babu (2015)	C	ocr	51000	(28, 28)
2015	MIT Scenes MI66	Li et al. (2015a)	C	scene	4554	(413, 501)
2015	SUN 397 Xiao et al. (2010),SU26,Xiao et al. (2016)	Li et al. (2015b)	M	scene	76128	(291, 353)
2015	CIFAR 10 Krizhevsky et al. (2009),CI25	He et al. (2016)	C	object	42500	(32, 32)

Year	Dataset Name	Sampled Paper	Type	Domain	Size	Avg. res.
2016	CUB 200 Wah et al. (2011),CU29	Moghimi et al. (2016)	C	object	5094	(386, 467)
2016	Stanford Cars Krause et al. (2013),St34	Moghimi et al. (2016)	C	object	6937	(483, 700)
2016	FGVC Aircraft Maji et al. (2013),FG31	Moghimi et al. (2016)	C	object	5683	(747, 1099)
2016	DTD Cimpoi et al. (2014),DT30	Shahriari (2016)	C	texture	1880	(451, 496)
2016	MS COCO Lin et al. (2014),MS32	Zhao et al. (2016)	C	object	82783	(130, 127)
2016	Caltech 101 Li et al. (2022),Ca16	Guo and Lew (2016)	C	object	2601	(244, 301)
2016	Oxford IIIT Pets Parkhi et al. (2012),Ox33	Kobayashi (2016)	C	object	3128	(391, 437)
2016	Stanford Dogs Khosla et al. (2011),St35	Kobayashi (2016)	C	object	10200	(385, 442)
2016	ANIMAL Bai et al. (2009),AN27	Li et al. (2016)	C	shape	1346	(474, 581)
2016	Caltech 101 Silhouettes Marlin et al. (2010),Ca28	Li et al. (2016)	C	shape	4100	(28, 28)
2016	Interact Antol et al. (2014),In39	Sharmanska and Quadrianto (2016)	C	object	2090	(475, 582)
2016	VOC Actions Everingham et al. (2010),VO37	Aljundi et al. (2017)	M	object	4234	(226, 150)
2016	SVHN Netzer et al. (2011),SV36	Aljundi et al. (2017)	C	object	62268	(32, 32)
2016	Chars74K de Campos et al. (2009),Ch38	Aljundi et al. (2017)	C	ocr	45740	(158, 168)
2017	CelebA Liu et al. (2015),Ce40	Kalayeh et al. (2017)	C	face	162770	(218, 178)
2017	LFWA Taigman et al. (2009),LF42	Kalayeh et al. (2017)	C	face	4716	(250, 250)
2017	SUN Attribute Patterson and Hays (2012),SU43,Patterson et al. (2014)	Kodirov et al. (2017)	M	scene	9692	(478, 580)
2017	CIFAR 100 Krizhevsky (2009a),CI41	Rebuffi et al. (2017b)	C	object	42500	(32, 32)
2018	TID2008 Ponomarenko et al. (2009),TI50	Lin and Wang (2018)	C	quality	1149	(384, 512)
2018	TID2013 Ponomarenko et al. (2015),TI51	Lin and Wang (2018)	C	quality	1973	(384, 512)
2018	USPS Hull (1994),US52	Liu et al. (2018)	C	ocr	6207	(16, 16)
2018	Semeion Buscema (1998),Se49,Dua and Graff (2017)	Liu et al. (2018)	C	ocr	1074	(16, 16)
2018	MNIST-m Ganin and Lempitsky (2015),MN46	Mancini et al. (2018)	C	ocr	46111	(32, 32)
2018	Office Caltech Gong et al. (2012),Of47	Mancini et al. (2018)	C	object	1410	(360, 373)
2018	PACS Li et al. (2017),PA48	Mancini et al. (2018)	C	object	6062	(227, 227)
2018	Caltech Camera Traps Beery et al. (2018a),Ca44	Beery et al. (2018b)	C	object	13553	(748, 1024)

Year	Dataset Name	Sampled Paper	Type	Domain	Size	Avg. res.
2018	EMNIST Balanced Cohen et al. (2017),EM45	Jayaraman et al. (2018)	C	ocr	95880	(28, 28)
2018	CASIA-HWDB1.1 Liu et al. (2011),CA67	Jayaraman et al. (2018)	C	ocr	797600	(81, 70)
2018	ISBI-ISIC 2017 melanoma classification challenge Codella et al. (2018),IS68	Radhakrishnan et al. (2018)	C	medical	2000	(2228, 3281)
2019	Trancos Guerrero-Gómez-Olmedo et al. (2015),Tr79	Hossain et al. (2019)	C	counting	403	(480, 640)
2019	Mall dataset Chen et al. (2012),Ma74,Change Loy et al. (2013),Chen et al. (2013),Loy et al. (2013)	Hossain et al. (2019)	C	counting	680	(480, 640)
2019	Fashion MNIST Xiao et al. (2017),Fa72	Shafaei et al. (2019)	C	object	51000	(28, 28)
2019	NotMNIST Bulatov (2011),No76	Shafaei et al. (2019)	C	ocr	12345	(28, 28)
2019	Tiny Imagenet mnmoustafa (2017),Ti78	Shafaei et al. (2019)	C	object	85099	(64, 64)
2019	STL10 Coates et al. (2011),ST77	Shafaei et al. (2019)	C	object	4250	(96, 96)
2019	Food 101 Bossard et al. (2014),Fo73 (a)	Tan and Le (2019)	C	object	75750	(475, 495)
2019	MNIST-rot Larochelle et al. (2007),MN75	Murugan et al. (2019)	C	ocr	51104	(28, 28)
2019	AWA2 Xian et al. (2018),AW71	Elhoseiny and Elfeki (2019)	M	object	25827	(192, 245)
2019	15 Scenes Lazebnik et al. (2006b),Sc70	Jiang et al. (2019)	C	scene	3021	(244, 273)
2019	Food 101n Lee et al. (2018),Fo73 (b)	Zhang et al. (2019b)	C	object	45032	(361, 394)
2019	COIL 20 Sameer et al. (1996a),CO8	Zhang et al. (2019a)	C	object	973	(128, 128)
2019	COIL 100 Sameer et al. (1996b),CO10	Zhang et al. (2019a)	C	object	6120	(128, 128)
2020	ShanghaiTech Zhang et al. (2016),Sh98	Duan et al.	C	counting	595	(696, 961)
2020	AnimalWeb Khan et al. (2020),An96	Khan et al. (2020)	C	object	12909	(1154, 1461)
2020	BIWI Fanelli et al. (2013),BI97	Pan et al. (2020)	M	face	11325	(480, 640)
2021	ImageNet Deng et al. (2009a),Im90	Pi et al. (2021b)	C	object	1281167	(406, 473)
2021	Oxford Flowers 102 Nilsback and Zisserman (2008),Ox94	Pi et al. (2021a)	C	object	1020	(534, 630)
2021	DomainNet-Real Peng et al. (2019),Do100	Peng et al. (2019)	C	object	102770	(467, 472)
2021	Pneumonia Chest X-ray Kermany et al. (2018),Pn104	Kothawade et al. (2022)	C	xray	5216	(970, 1327)
2021	Path MNIST Kather et al. (2016),Pa103	Kothawade et al. (2022)	C	medical	3356	(28, 28)
2021	NIH Chest X-ray Wang et al. (2017),NI101	Tetteh et al. (2021)	M	xray	73638	(1024, 1024)

Year	Dataset Name	Sampled Paper	Type	Domain	Size	Avg. res.
2021	Tubercolosis Rahman et al. (2020),Tu106	Cherti et al. (2021)	C	xray	2809	(512, 512)
2021	covid-19 x-ray Chowdhury et al. (2020),co99,Rahman et al. (2021)	Cherti and Jitsev (2022)	C	xray	14281	(299, 299)
2021	PatchCamelyon Veeling et al. (2018),Pa102	Yang (2021)	C	medical	262144	(96, 96)
2021	DDSM Heath et al. (2001); Lee et al. (2017)	Yang (2021)	C	medical	2075	(5253, 3116)
2021	Synthetic COVID-19 Chest X-ray Dataset Zunair and Hamza (2021),Sy105	Zunair and Hamza (2021)	C	xray	14410	(256, 256)

Appendix M. List of Datasets in the short version of Nevis'22

Year	Dataset Name	Type	Domain	Size	Avg. res.
2004	COIL 100 Nayar (1996),CO10	C	object	6120	(128, 128)
2004	MNIST LeCun et al. (1998a),MN14	C	ocr	51000	(28, 28)
2006	Pascal 2005 Everingham et al. (2005),Pa23	C	object	881	(430, 553)
2006	Caltech Categories Fergus et al. (2003),Ca22	C	object	996	(341, 514)
2006	UIUC cars Agarwal et al. (2004),UI24	C	object	823	(50, 112)
2009	Pascal 2006 Everingham et al. (2006),Pa83	C	object	2211	(420, 524)
2010	Caltech 101 Li et al. (2022),Ca16	C	object	2601	(244, 301)
2011	Graz-02 Opelt et al. (2006),Gr84,Marszalek and Schmid (2007)	C	object	747	(497, 622)
2011	15 Scenes Lazebnik et al. (2006b),Sc70	C	scene	3021	(244, 273)
2011	Pascal 2007 Pa85	M	object	2501	(382, 471)
2011	LFW Huang et al. (2007),LF92	C	face	11248	(250, 250)
2013	sketch dataset Eitz et al. (2012),sk63	C	object	13536	(1111, 1111)
2013	Brodatz Br3	C	texture	672	(213, 213)
2014	ImageNet Deng et al. (2009a),Im90	C	object	1281167	(406, 473)
2014	Pascal 2012 Pa64	C	object	5717	(386, 470)
2014	Caltech 256 Griffin et al. (2022),Ca88	C	object	20696	(325, 371)
2018	CIFAR 100 Krizhevsky (2009a),CI41	C	object	42500	(32, 32)
2018	CIFAR 10 Krizhevsky et al. (2009),CI25	C	object	42500	(32, 32)
2018	USPS Hull (1994),US52	C	ocr	6207	(16, 16)
2018	MNIST LeCun et al. (1998a),MN14	C	ocr	51000	(28, 28)
2018	MNIST-m Ganin and Lempitsky (2015),MN46	C	ocr	46111	(32, 32)
2018	Office Caltech Gong et al. (2012),Of47	C	object	1410	(360, 373)
2018	PACS Li et al. (2017),PA48	C	object	6062	(227, 227)
2018	ISBI-ISIC 2017 melanoma classification challenge Codella et al. (2018),IS68	C	medical	2000	(2228, 3281)
2019	Fashion MNIST Xiao et al. (2017),Fa72	C	object	51000	(28, 28)
2020	Stanford Dogs Khosla et al. (2011),St35	C	object	10200	(385, 442)
2020	CUB 200 Wah et al. (2011),CU29	C	object	5094	(386, 467)
2020	Stanford Cars Krause et al. (2013),St34	C	object	6937	(483, 700)
2020	FGVC Aircraft Maji et al. (2013),FG31	C	object	5683	(747, 1099)

Appendix N. Individual results on Nevis'22

In this section, we provide the performance of different learners on individual datasets during their training on the full stream. These results correspond to the regret plot 5 from the main paper. For reference, we provide the cumulative training FLOPs and the test average errors corresponding the reported learners in the first table, while the second table provide the accuracy or the mean average precision on the test sets. In this table, we also mark the meta-train and meta-test streams, and highlight the repeated datasets (in bold).

Method	Cumulative training FLOPs	Average error (%)
Indep	1.3e+20	27.56
FT: previous	1.3e+20	25.97
FT: Dynamic kNN	1.3e+20	23.18
MT	1.9e+20	23.32

Dataset	Indep	FT: previous	FT: Dynamic kNN	MT
Meta-Train				
magellan venus volcanoes	66.67	66.67	72.22	61.11
aberdeen	95.97	95.16	95.16	94.35
landsat	90.25	90.40	90.90	92.20
brodatz	95.09	95.54	90.18	91.96
olivetti face	98.39	96.77	100.00	98.39
coil20	100.00	100.00	100.00	100.00
coil100	92.96	88.58	92.42	92.25
mpeg7	87.02	87.40	83.59	87.02
butterflies	97.14	91.43	94.29	95.71
mnist	99.41	99.26	99.30	99.32
caltech101	63.28	69.89	66.78	66.90
umist	100.00	100.00	99.46	100.00
cmu amp expression	100.00	100.00	99.52	99.04
pascal voc2005	62.36	60.87	60.60	61.55
caltech categories	100.00	100.00	100.00	100.00
uiuc cars	98.70	98.70	98.26	98.70
aloi	84.28	88.61	85.37	82.91
scenes8	89.15	91.54	91.73	93.01
scenes15	84.82	86.14	84.93	84.82
pascal voc2006	64.03	67.42	67.16	67.09
extended yaleb	99.30	99.55	98.86	99.19
voc 2007	48.46	48.95	45.83	47.51
ig02	92.50	94.50	95.00	93.00
olivetti face	96.77	100.00	100.00	98.39
ppmi	45.17	40.92	43.25	38.79
caltech256	58.25	59.46	58.46	58.65
imagenet2012	62.06	63.16	62.83	62.86
lfw	44.68	51.30	50.24	52.25
oxford flowers 17	90.00	93.82	97.06	96.76
flickr material database	35.16	45.05	72.53	70.33
oxford flowers102	54.77	73.26	84.52	85.75
belgium tsc	97.46	97.14	96.94	98.13
german tsr	96.88	96.86	96.19	96.17
brodatz	95.54	91.96	93.75	95.09

Dataset	Indep	FT: previous	FT: Dynamic kNN	MT
vistex	39.13	56.52	69.57	56.52
umd	98.35	98.90	99.45	98.35
kth tips	100.00	100.00	99.31	98.62
uiuc texture	99.45	97.25	97.25	96.70
kth tips 2b	99.59	99.18	99.39	99.28
cvc muscima	14.60	12.60	21.20	14.40
sketch	61.96	60.75	61.30	62.72
kth tips 2a	99.36	99.68	99.89	99.58
smallnorb	74.67	79.88	84.00	86.89
wiki paintings style	55.49	54.50	58.08	59.43
mnist	99.41	99.20	99.24	99.32
mit scenes	48.96	51.04	66.19	68.43
sun397	60.44	58.62	65.17	65.41
cifar10	93.02	94.11	96.35	95.93
caltech birds2011	46.60	56.96	67.47	67.45
stanford cars	62.16	75.00	80.13	80.96
fgvc aircraft family	74.95	73.57	80.14	80.35
dtd	38.72	43.30	55.53	56.01
coco single label	75.56	75.24	77.50	77.17
caltech101	64.04	75.90	84.35	83.12
oxford iiit pet	64.84	72.42	85.91	85.83
stanford dogs	49.04	56.41	72.88	71.70
animal	88.61	87.09	84.30	88.10
silhouettes 28	7.28	14.65	17.47	12.70
interact	12.14	12.55	15.83	21.15
voc actions	52.31	54.07	46.05	70.59
svhn cropped	95.94	96.02	95.59	95.52
chars74k	86.00	86.84	79.54	86.64
celeb a	79.04	78.67	78.96	78.42
lfwa	48.70	55.79	53.66	59.22
sun attributes	55.53	50.03	61.17	60.52
cifar100	73.77	74.14	82.03	81.48
tid2008	46.44	42.72	48.92	40.56
tid2013	53.06	50.58	55.37	55.87
usps	96.81	96.81	97.16	97.41
semeion	95.02	94.35	97.67	96.35
mnist m	86.57	86.21	86.27	85.99
office caltech 10	58.28	64.14	85.52	67.59
pacs	27.41	44.94	57.18	47.10
caltech camera traps	31.27	30.93	31.76	34.14
emnist balanced	89.47	88.99	89.29	89.13
casia hwdb	93.41	93.72	93.66	93.53
melanoma	69.00	70.83	73.83	76.67
trancos	12.35	13.54	14.96	13.78
Meta-Test				
mall	14.42	16.75	13.92	14.33
fashion mnist	94.49	94.01	94.46	94.33
not mnist	95.23	95.39	95.58	95.55
tiny imagenet	51.76	48.90	68.88	66.71
stl10	80.96	84.46	95.25	95.19
food101	75.14	71.79	78.06	77.70
mnist rotation	96.08	95.81	96.30	96.58
awa2	97.80	97.76	98.83	99.27

Dataset	Indep	FT: previous	FT: Dynamic kNN	MT
scenes15	84.49	83.72	84.93	85.15
food101n	51.99	50.62	59.42	59.71
coil20	100.00	100.00	100.00	100.00
coil100	91.40	92.42	91.75	94.21
shanghai tech	29.12	27.11	27.71	27.71
animal web	53.22	60.31	71.03	71.83
biwi	79.58	71.28	70.97	70.66
imagenet2012	60.88	63.62	65.44	65.49
oxford flowers102	55.26	85.64	84.68	85.77
domainnet	31.15	31.91	35.70	35.51
pneumonia chest xray	94.87	88.62	92.31	90.38
path mnist	95.42	95.42	94.74	95.42
nih chest xray	22.51	18.99	21.65	21.04
tubercolosis	98.61	98.84	98.84	99.54
covid 19 xray	93.56	93.49	94.24	94.05
patch camelyon	71.33	71.37	71.28	71.04
ddsm	48.37	51.63	49.12	51.63
synthetic covid19 xray	100.00	100.00	99.98	99.95

References

Ab1. Dataset: 'aberdeen face database.' link. URL http://pics.stir.ac.uk/2D_face_sets.htm.

Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE transactions on pattern analysis and machine intelligence*, 26(11):1475–1490, 2004.

AL20. Dataset: 'aloi' link. URL https://aloi.science.uva.nl/.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3366–3375, 2017.

AN27. Dataset: 'animal' link. URL https://sites.google.com/site/xiangbai/animaldataset.

An96. Dataset: 'animalweb' link. URL https://fdmaproject.wordpress.com/author/fdmaproject/.

Stanislaw Antol, C. Lawrence Zitnick, and Devi Parikh. Zero-Shot Learning via Visual Abstraction. In *ECCV*, 2014.

Randy Ardywibowo, Shahin Boluki, Xinyu Gong, Zhangyang Wang, and Xiaoning Qian. Nads: Neural architecture distribution search for uncertainty awareness. In *International Conference on Machine Learning*, pages 356–366. PMLR, 2020.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In *Neural Information Processing Systems*, 2020.

AW71. Dataset: 'awa2' link. URL https://cvml.ist.ac.at/AwA2/.

Haoyue Bai, Fengwei Zhou, Lanqing Hong, Nanyang Ye, S-H Gary Chan, and Zhenguo Li. Nas-ood: Neural architecture search for out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8320–8329, 2021.

Xiang Bai, Wenyu Liu, and Zhuowen Tu. Integrating contour and skeleton for shape classification. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pages 360–367. IEEE, 2009.

Evgeniy Bart and Shimon Ullman. Single-example learning of novel classes using representation by similarity. In *BMVC*, volume 1, page 2. Citeseer, 2005.

Be87. Dataset: 'belgium traffic sign dataset' link. URL http://people.ee.ethz.ch/~timofter/traffic_signs/.

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI*, pages 472–489, 2018a. doi: 10.1007/978-3-030-01270-0_28. URL https://doi.org/10.1007/978-3-030-01270-0_28.

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018b.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham Taylor, and Daniel Silver, editors, *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27 of *Proceedings of Machine Learning Research*, pages 17–36, Bellevue, Washington, USA, 02 Jul 2012. PMLR. URL https://proceedings.mlr.press/v27/bengio12a.html.

BI97. Dataset: 'biwi' link. URL https://www.kaggle.com/kmader/biwi-kinect-head-pose-database.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

Br3. Dataset: 'brodatz' link. URL https://sipi.usc.edu/database/database.php?volume=textures.

Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pages 1059–1071. PMLR, 2021.

Phil Brodatz. *Textures: a photographic album for artists and designers*. New York: Dover Pub., 1966.

Martin Brown, Nicholas Paul Costen, and Chester Street. Non-linear feature selection for classification. In *BMVC*, pages 1–10, 2004.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Bu13. Dataset: 'butterfly dataset' link. URL https://web.archive.org/web/20070829035029/http://www-cvr.ai.uiuc.edu/ponce_grp/data/index.html.

Yaroslav Bulatov. Notmnist dataset. *Google (Books/OCR), Tech. Rep.[Online]. Available: http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html*, 2, 2011.

Massimo Buscema. Metanet*: The theory of independent judges. *Substance Use & Misuse*, 33(2):439–461, 1998. doi: 10.3109/10826089809115875. URL https://doi.org/10.3109/10826089809115875. PMID: 9516737.

Ca16. Dataset: 'caltech 101' link. URL https://data.caltech.edu/records/mzrjq-6wc02.

Ca22. Dataset: 'caltech categories' link. URL https://data.caltech.edu/records/pxb2q-1e144.

Ca28. Dataset: 'caltech 101 silhouettes' link. URL https://people.cs.umass.edu/~marlin/data.shtml.

Ca44. Dataset: 'caltech camera traps' link. URL https://beerys.github.io/CaltechCameraTraps/#:~:text=Caltech%20Camera%20Traps%20(CCT),approximately%20one%20frame%20per%20second.

CA67. Dataset: 'casia-hwdb1.1' link. URL http://www.nlpr.ia.ac.cn/databases/handwriting/Offline_database.html.

Ca88. Dataset: 'caltech 256' link. URL https://data.caltech.edu/records/nyy15-4j048.

Lucas Caccia, Jing Xu, Myle Ott, Marc'Aurelio Ranzato, and Ludovic Denoyer. On anytime learning at macroscale. In *Conference on Lifelong Learning Agents*, 2022.

Hongping Cai, Krystian Mikolajczyk, and Jiri Matas. Learning linear discriminant projections for dimensionality reduction of image descriptors. In *Proc. BMVC*, 2008.

Zhipeng Cai, Ozan Sener, and Vladlen Koltun. Online continual learning with natural distribution shifts: An empirical study with visual data. In *International Conference on Computer Vision*, 2021. URL https://arxiv.org/abs/2108.09020.

Ce40. Dataset: 'celeba' link. URL https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html.

Ch38. Dataset: 'chars74k' link. URL http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/.

Chen Change Loy, Shaogang Gong, and Tao Xiang. From semi-supervised to transfer counting of crowds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2256–2263, 2013.

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets, 2014. URL https://arxiv.org/abs/1405.3531.

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, volume 11215, pages 556–572, 2018.

Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *Bmvc*, volume 1, page 3, 2012.

Ke Chen, Shaogang Gong, Tao Xiang, and Chen Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2467–2474, 2013.

Liang Chen. Pairwise macropixel comparison can work at least as well as advanced holistic algorithms for face recognition. In *Proc. BMVC*, pages 5.1–11, 2010. ISBN 1-901725-40-5. doi:10.5244/C.24.5.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

Mehdi Cherti and Jenia Jitsev. Effect of pre-training scale on intra- and inter-domain, full and few-shot transfer learning for natural and x-ray chest images. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, 2022. doi: 10.1109/IJCNN55064.2022.9892393.

Mehdi Cherti, Jenia Jitsev, and AI Helmholtz. Effect of pre-training scale on intra-and inter-domain transfer for natural and x-ray chest images. In *Medical Imaging meets NeurIPS workshop*, 2021.

Muhammad E. H. Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, and Mohammad Tariqul Islam. Can ai help in screening viral and covid-19 pneumonia? *IEEE Access*, 8:132665–132676, 2020. doi: 10.1109/ACCESS.2020.3010287.

CI25. Dataset: 'cifar 10' link. URL `https://www.cs.toronto.edu/~kriz/cifar.html`.

CI41. Dataset: 'cifar 100' link. URL `https://www.cs.toronto.edu/~kriz/cifar.html`.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

CM18. Dataset: 'cmu amp expression' link. URL `http://chenlab.ece.cornell.edu/projects/FaceAuthentication/download.html`.

CO10. Dataset: 'coil 100' link. URL https://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php.

CO8. Dataset: 'coil 20' link. URL https://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php.

co99. Dataset: 'covid-19 x-ray' link. URL https://www.kaggle.com/tawsifurrahman/covid19-radiography-database.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 168–172. IEEE, 2018.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017. doi: 10.1109/ijcnn.2017.7966217.

Ian Craw and Peter Cameron. Face recognition by computer. In *BMVC*, 1992.

CU29. Dataset: 'cub 200' link. URL http://www.vision.caltech.edu/datasets/cub_200_2011/.

CV55. Dataset: 'cvc-muscima' link. URL http://www.cvc.uab.es/cvcmuscima/index_database.html.

Anders Lindbjerg Dahl and Rasmus Larsen. Learning dictionaries of discriminative image patches. In *BMVC*, pages 1–11, 2011.

T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In *Proceedings of the International Conference on Computer Vision Theory and Applications, Lisbon, Portugal*, February 2009.

Vincent Delaitre, Ivan Laptev, and Josef Sivic. Recognizing human actions in still images: a study of bag-of-features and part-based representations. In *BMVC 2010-21st British Machine Vision Conference*, 2010.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009a. doi: 10.1109/CVPR.2009.5206848.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009b. doi: 10.1109/CVPR.2009.5206848.

Liefeng Bo Diane Hu and Xiaofeng Ren. Toward robust material recognition for everyday objects. In *Proceedings of the British Machine Vision Conference*, pages 48.1–48.11. BMVA Press, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.48.

Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, 2000.

Do100. Dataset: 'domainnet-real' link. URL `http://ai.bu.edu/M3SDA/`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

DT30. Dataset: 'dtd' link. URL `https://www.robots.ox.ac.uk/~vgg/data/dtd/`.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Haoran Duan, Shidong Wang, and Yu Guan. Sofa-net: Second-order and first-order attention network for crowd counting.

Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

Mohamed Elhoseiny and Mohamed Elfeki. Creativity inspired zero-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5784–5793, 2019.

EM45. Dataset: 'emnist balanced' link. URL `https://www.nist.gov/itl/products-and-services/emnist-dataset`.

Hugo Jair Escalante, Carlos A. Hernández, Jesus A. Gonzalez, A. López-López, Manuel Montes, Eduardo F. Morales, L. Enrique Sucar, Luis Villaseñor, and Michael Grubinger. The segmented and annotated iapr tc-12 benchmark. *Computer Vision and Image Understanding*, 114(4):419–428, 2010. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2009.03.008. URL `https://www.sciencedirect.com/science/article/pii/S1077314209000575`. Special issue on Image and Video Retrieval Evaluation.

Mark Everingham, Andrew Zisserman, Christopher Williams, Luc Van Gool, Moray Allan, Christopher Bishop, Olivier Chapelle, Navneet Dalal, Thomas Deselaers, Gyuri Dorkó, Stefan Duffner, Jan Eichhorn, Jason Farquhar, Mario Fritz, Christophe Garcia, Tom Griffiths, Frederic Jurie, Daniel Keysers, Markus Koskela, and Jianguo Zhang. The 2005 pascal visual object classes challenge. volume 3944, pages 117–176, 04 2005. ISBN 978-3-540-33427-9. doi: 10.1007/11736790_8.

Mark Everingham, Andrew Zisserman, Christopher K. I. Williams, and Luc Van Gool. The pascal visual object classes challenge 2006 (voc2006) results. 2006.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

Ex82. Dataset: 'extended yaleb' link. URL `http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html`.

Fa72. Dataset: 'fashion mnist' link. URL `https://www.tensorflow.org/datasets/catalog/fashion_mnist`.

Gabriele Fanelli, Matthias Dantone, Juergen Gall, Andrea Fossati, and Luc Van Gool. Random forests for real time 3d face analysis. *Int. J. Comput. Vision*, 101(3):437–458, February 2013.

Robert Fergus, Pietro Perona, and Andrew Zisserman. Object class recognition by unsupervised scale-invariant learning. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–II. IEEE, 2003.

Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. Initializing bayesian hyperparameter optimization via meta-learning. In *AAAI Conference on Artificial Intelligence*, 2015.

FG31. Dataset: 'fgvc aircraft' link. URL `https://www.robots.ox.ac.uk/~vgg/data/fgvc-aircraft/`.

Chelsea B Finn. *Learning to learn with gradients*. University of California, Berkeley, 2018.

Fl89. Dataset: 'flicker material dataset' link. URL `https://people.csail.mit.edu/lavanya/fmd.html`.

Fo73. Dataset: 'food 101' link, a. URL `https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/`.

Fo73. Dataset: 'food 101' link, b. URL `https://kuanghuei.github.io/Food-101N/`.

Alicia Fornés, Anjan Dutta, Albert Gordo, and Josep Lladós. Cvc-muscima: a ground truth of handwritten music score images for writer identification and staff removal. *International Journal on Document Analysis and Recognition (IJDAR)*, 15(3):243–251, 2012.

S. R. Fountain, T. N. Tan, and K. D. Baker. A comparative study of rotation invariant classification and retrieval of texture images. In *Proceedings of the British Machine Vision Conference*, pages 27.1–27.10. BMVA Press, 1998. ISBN 1-901725-04-9. doi:10.5244/C.12.27.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

Ge53. Dataset: 'german traffic sign recognition benchmark' link. URL `https://benchmark.ini.rub.de/gtsrb_news.html`.

Andrea Gesmundo and Jeff Dean. munet: Evolving pretrained deep neural networks into scalable auto-tuning multitask systems. *arXiv:2205.10937*, 2022.

Jan-Mark Geusebroek. Compact object descriptors from local colour invariant histograms. In *BMVC*, pages 1029–1038, 2006.

Jan-Mark Geusebroek, Gertjan J Burghouts, and Arnold WM Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

S. Gibson and R. W. Harvey. Recognition and retrieval via histogram trees. pages 531–540, September 2001. The British Machine Vision Conference ; Conference date: 10-09-2001 Through 13-09-2001.

Gurman Gill and Martin Levine. A single classifier for view-invariant multiple object class recognition. In *BMVC*, pages 257–266, 2006.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. Google Vizier: A service for black-box optimization. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1487–1495, 2017.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2066–2073. IEEE, 2012.

Gr84. Dataset: 'graz-02' link. URL `http://lear.inrialpes.fr/people/marszalek/data/ig02/`.

Daniel B Graham and Nigel M Allinson. Characterising virtual eigensignatures for general purpose face recognition. In *Face Recognition*, pages 446–456. Springer, 1998.

Griffin, Holub, and Perona. Caltech 256, Apr 2022.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Neural Information Processing Systems*, 2020.

Sam Gross, Marc'Aurelio Ranzato, and Arthur Szlam. Hard mixtures of experts for large scale weakly supervised vision. In *Computer Vision and Pattern Recognition*, 2017.

Ricardo Guerrero-Gómez-Olmedo, Beatriz Torre-Jiménez, Roberto López-Sastre, Saturnino Maldonado-Bascón, and Daniel O noro Rubio. Extremely overlapping vehicle counting. In *Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*, 2015.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

Yanming Guo and Michael S Lew. Bag of surrogate parts: one inherent feature of deep cnns. In *BMVC*, 2016.

R. Hadsell, D Rao, A.A. Rusu, and R. Pascanu. Embracing change: Continual learning in deep neural networks. 24(12):1028–1040, 2020.

Peter M. Hall, A. David Marshall, and Ralph Robert Martin. Incremental eigenanalysis for classification. In *BMVC*, 1998.

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.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

Michael Heath, Kevin Bowyer, Daniel Kopans, Richard Moore, and W. Philip Kegelmeyer. The digital database for screening mammography. pages 212–218. Medical Physics Publishing, 2001.

Andre Hegerath, Thomas Deselaers, and Hermann Ney. Patch-based object recognition using discriminatively trained gaussian mixtures. In *BMVC*, pages 519–528, 2006.

T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(09): 5149–5169, sep 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3079209.

Mohammad Asiful Hossain, Mahesh Kumar, Mehrdad Hosseinzadeh, Omit Chanda, and Yang Wang. One-shot scene-specific crowd counting. In *BMVC*, page 217, 2019.

Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

IA60. Dataset: 'iaprtc-12' link. URL `https://www.kaggle.com/datasets/nastyatima/iapr-tc12`.

Im90. Dataset: 'imagenet' link. URL `https://www.image-net.org/`.

In39. Dataset: 'interact' link. URL `https://computing.ece.vt.edu/~santol/projects/zsl_via_visual_abstraction/interact/index.html`.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/ioffe15.html`.

IS68. Dataset: 'isbi-isic 2017 melanoma classification challenge' link. URL `https://challenge.isic-archive.com/data/`.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, 2021.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *CoRR*, abs/2011.00362, 2020. URL https://arxiv.org/abs/2011.00362.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. Temporalwiki: A lifelong benchmark for training and evaluating ever-evolving language models, 2022. URL https://arxiv.org/abs/2204.14211.

Pradeep Kumar Jayaraman, Jianhan Mei, Jianfei Cai, and Jianmin Zheng. Quadtree convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 546–561, 2018.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11313–11320, 2019.

Gareth Jones. Genetic and evolutionary algorithms. *Encyclopedia of Computational Chemistry*, 2:1127–1136, 1998.

John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Zídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andy Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David A. Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583 – 589, 2021.

Mahdi M Kalayeh, Boqing Gong, and Mubarak Shah. Improving facial attribute prediction using semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6942–6950, 2017.

Sergey Karayev, Aaron Hertzmann, Matthew Trentacoste, Helen Han, Holger Winnemoeller, Aseem Agarwala, and Trevor Darrell. Recognizing image style. In *Proceedings of the British Machine Vision Conference 2014*. British Machine Vision Association, 2014. doi: 10.5244/c.28.122. URL https://doi.org/10.5244%2Fc.28.122.

Jakob Nikolas Kather, Frank Gerrit Zöllner, Francesco Bianconi, Susanne M Melchers, Lothar R Schad, Timo Gaiser, Alexander Marx, and Cleo-Aron Weis. Collection of textures in colorectal cancer histology, May 2016. URL https://doi.org/10.5281/zenodo.53169.

Daniel S Kermany, Michael Goldbaum, Wenjia Cai, Carolina CS Valentim, Huiying Liang, Sally L Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5): 1122–1131, 2018.

Muhammad Haris Khan, John McDonagh, Salman Khan, Muhammad Shahabuddin, Aditya Arora, Fahad Shahbaz Khan, Ling Shao, and Georgios Tzimiropoulos. Animalweb: A large-scale hierarchical dataset of annotated animal faces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6939–6948, 2020.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. 2020.

Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

Byung Soo Kim, Jae Young Park, Anush Mohan, Anna Gilbert, and Silvio Savarese. Hierarchical classification of images by sparse approximation. In *Proceedings of the British Machine Vision Conference*, pages 106.1–106.11. BMVA Press, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.106.

Takumi Kobayashi. Learning additive kernel for feature transformation and its application to cnn features. In *BMVC*, 2016.

Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3174–3183, 2017.

Hui Kong, Xuchun Li, Jian-Gang Wang, Eam Khwang Teoh, and Chandra Kambhamettu. Discriminant low-dimensional subspace analysis for face recognition with small number of training samples. In *BMVC*, 2005.

Suraj Kothawade, Atharv Savarkar, Venkat Iyer, Ganesh Ramakrishnan, and Rishabh Iyer. Clinical: Targeted active learning for imbalanced medical image classification. In *Workshop on Medical Image Learning with Limited and Noisy Data*, pages 119–129. Springer, 2022.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009a.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto, technical report*, 2009b.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

KT56. Dataset: 'kth-tips' link. URL https://www.csc.kth.se/cvap/databases/kth-tips/download.html.

KT57. Dataset: 'kth-tips2-b' link. URL https://www.csc.kth.se/cvap/databases/kth-tips/download.html.

KT61. Dataset: 'kth-tips2-a' link. URL https://www.csc.kth.se/cvap/databases/kth-tips/index.html.

La5. Dataset: 'landsat uci repo' link. URL https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite).

B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480, 2007.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178, 2006a. doi: 10.1109/CVPR.2006.68.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Semi-local Affine Parts for Object Recognition. In Andreas Hoppe, Sarah Barman, and Tim Ellis, editors, *British Machine Vision Conference (BMVC '04)*, pages 779–788, Kingston, United Kingdom, September 2004. The British Machine Vision Association (BMVA). URL https://hal.inria.fr/inria-00548542.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A sparse texture representation using local affine regions. *IEEE transactions on pattern analysis and machine intelligence*, 27(8): 1265–1278, 2005.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006b.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998a.

Yann LeCun, Leon Bottou, and and Patrick Haffner Yoshua Bengio. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998b.

Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–104. IEEE, 2004.

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

Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Rebecca Sawyer Lee, Francisco Gimenez, Assaf Hoogi, Kanae Kawai Miyake, Mia Gorovoy, and Daniel L Rubin. A curated mammography data set for use in computer-aided detection and diagnosis research. *Scientific data*, 4(1):1–9, 2017.

LF42. Dataset: 'lfwa' link. URL `https://talhassner.github.io/home/projects/lfwa/`.

LF92. Dataset: 'lfw' link. URL `http://vis-www.cs.umass.edu/lfw/`.

Li, Andreeto, Ranzato, and Perona. Caltech 101, Apr 2022.

Chunyuan Li, Andrew Stevens, Changyou Chen, Yunchen Pu, Zhe Gan, and Lawrence Carin. Learning weight uncertainty with stochastic gradient mcmc for shape classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5666–5675, 2016.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision*, 2017.

Peihua Li, Xiaoxiao Lu, and Qilong Wang. From dictionary of visual words to subspaces: Locality-constrained affine subspace coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015a.

Peihua Li, Xiaoxiao Lu, and Qilong Wang. From dictionary of visual words to subspaces: Locality-constrained affine subspace coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2348–2357, 2015b.

Yi Li, Yi-Zhe Song, Shaogang Gong, et al. Sketch recognition by ensemble matching of structured features. In *BMVC*, volume 1, page 2, 2013.

Kwan-Yee Lin and Guanxiang Wang. Self-supervised deep multiple choice learning network for blind image quality assessment. In *BMVC*, page 70, 2018.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014. URL `https://arxiv.org/abs/1405.0312`.

Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The CLEAR benchmark: Continual LEArning on real-world imagery. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL `https://openreview.net/forum?id=43mYF598ZDB`.

Marius Lindauer and Frank Hutter. Warmstarting of model-based algorithm configuration. In *AAAI*, 2018.

Adam Liska, Tomas Kocisky, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, D'Autume Cyprien De Masson, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. In *International Conference on Machine Learning*, pages 13604–13622, 2022.

Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Casia online and offline chinese handwriting databases. In *2011 International Conference on Document Analysis and Recognition*, pages 37–41. IEEE, 2011.

Yen-Cheng Liu, Yu-Ying Yeh, Tzu-Chien Fu, Sheng-De Wang, Wei-Chen Chiu, and Yu-Chiang Frank Wang. Detach and adapt: Learning cross-domain disentangled deep representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8867–8876, 2018.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. *arXiv:1705.03550*, 2017.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6467–6476, 2017. URL `http://papers.nips.cc/paper/7225-gradient-episodic-memory-for-continual-learning`.

Chen Change Loy, Ke Chen, Shaogang Gong, and Tao Xiang. Crowd counting and profiling: Methodology and evaluation. In *Modeling, simulation and visual analysis of crowds*, pages 347–382. Springer, 2013.

Ma2. Dataset: 'magellan venus volcanoes' link. URL `http://archive.ics.uci.edu/ml/datasets/volcanoes+on+venus+-+jartool+experiment`.

Ma74. Dataset: 'mall dataset' link. URL `https://personal.ie.cuhk.edu.hk/~ccloy/downloads_mall_dataset.html`.

Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3771–3780, 2018.

Benjamin Marlin, Kevin Swersky, Bo Chen, and Nando Freitas. Inductive principles for restricted boltzmann machine learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 509–516. JMLR Workshop and Conference Proceedings, 2010.

Marcin Marszalek and Cordelia Schmid. Accurate object localization with shape masks. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

Jiri Matas, Jan Buriánek, and Josef Kittler. Object recognition using the invariant pixel-set signature. In *BMVC*, 2000.

Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition—how far are we from the solution? In *The 2013 international joint conference on Neural networks (IJCNN)*, pages 1–8. IEEE, 2013.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

Dong Xu Meina Kan, Shiguang Shan and Xilin Chen. Side-information based linear discriminant analysis for face recognition. In *Proceedings of the British Machine Vision Conference*, pages 125.1–125.0. BMVA Press, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.125.

MI66. Dataset: 'mit scenes' link. URL `https://www.kaggle.com/itsahmad/indoor-scenes-cvpr-2019`.

MN14. Dataset: 'mnist' link. URL `http://yann.lecun.com/exdb/mnist/`.

MN46. Dataset: 'mnist-m' link. URL `http://yaroslav.ganin.net/`.

MN75. Dataset: 'mnist-rot' link. URL `https://paperswithcode.com/task/rotated-mnist`.

Mohammed Ali mnmoustafa. Tiny imagenet visual recognition challenge. `https://tiny-imagenet.herokuapp.com/`, 2017. URL `https://kaggle.com/competitions/tiny-imagenet`.

Mohammad Moghimi, Serge J Belongie, Mohammad J Saberian, Jian Yang, Nuno Vasconcelos, and Li-Jia Li. Boosted convolutional neural networks. In *BMVC*, volume 5, page 6, 2016.

MP12. Dataset: 'mpeg-7' link. URL `https://dabi.temple.edu/external/shape/MPEG7/dataset.html`.

MS32. Dataset: 'ms coco' link. URL `https://cocodataset.org/#home`.

Krystian Mikolajczyk Muhammad Awais, Fei Yan and Josef Kittler. Augmented kernel matrix vs classifier fusion for object recognition. In *Proceedings of the British Machine Vision Conference*, pages 60.1–60.11. BMVA Press, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.60.

Muthuvel Murugan, KV Subrahmanyam, and Siruseri IT Park. So (2)-equivariance in neural networks using tensor nonlinearity. In *BMVC*, 2019.

Sheila J. Nayar. Columbia object image library (coil100). 1996.

Samer A. Nene, Shree K. Nayar, and Hiroshi Murase. Columbia object image library (coil-20). Technical Report CUCS-005-96, Department of Computer Science, Columbia University, February 1996.

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

NI101. Dataset: 'nih chest x-ray' link. URL `https://nihcc.app.box.com/v/ChestXray-NIHCC`.

M-E Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1447–1454. IEEE, 2006.

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.

NO62. Dataset: 'norb' link. URL `https://cs.nyu.edu/~ylclab/data/norb-v1.0/`.

No76. Dataset: 'notmnist' link. URL `http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html`.

Of47. Dataset: 'office caltech' link. URL `https://paperswithcode.com/dataset/office-caltech-10`.

Ol6. Dataset: 'olivetti face dataset' link. URL `https://www.kaggle.com/tavarez/the-orl-database-for-training-and-testing`.

A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 28(3), 2006.

Ox33. Dataset: 'oxford iiit pets' link. URL `https://www.robots.ox.ac.uk/~vgg/data/pets/#:~:text=Overview,and%20pixel%20level%20trimap%20segmentation.`

Ox93. Dataset: 'oxford flowers' link. URL `https://www.robots.ox.ac.uk/~vgg/data/flowers/17`.

Ox94. Dataset: 'oxford flowers 102' link. URL `https://www.robots.ox.ac.uk/~vgg/data/flowers/102/`.

Pa102. Dataset: 'patchcamelyon' link. URL `https://patchcamelyon.grand-challenge.org/`.

Pa103. Dataset: 'path mnist' link. URL `https://www.kaggle.com/datasets/kmader/colorectal-histology-mnist`.

Pa23. Dataset: 'pascal 2005' link. URL http://host.robots.ox.ac.uk/pascal/VOC/voc2005/index.html.

PA48. Dataset: 'pacs' link. URL https://dali-dl.github.io/project_iccv2017.html.

Pa64. Dataset: 'pascal 2012' link. URL http://host.robots.ox.ac.uk/pascal/VOC/voc2012/.

Pa83. Dataset: 'pascal 2006' link. URL http://host.robots.ox.ac.uk/pascal/VOC/databases.html#VOC2006.

Pa85. Dataset: 'pascal 2007' link. URL http://host.robots.ox.ac.uk/pascal/VOC/voc2007/.

Lili Pan, Shijie Ai, Yazhou Ren, and Zenglin Xu. Self-paced deep regression forests with consideration on underrepresented examples. In *European Conference on Computer Vision*, pages 271–287. Springer, 2020.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 2010.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54 – 71, 2019. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2019.01.012. URL http://www.sciencedirect.com/science/article/pii/S0893608019300231.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *Proceeding of the 25th Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Genevieve Patterson, Chen Xu, Hang Su, and James Hays. The sun attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*, 108(1-2):59–81, 2014.

Amandalynne Paullada, Inioluwa Deborah Raji, Emily M. Bender, Emily Denton, and Alex Hanna. Data and its (dis)contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11), 2021.

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.

Huaijin Pi, Huiyu Wang, Yingwei Li, Zizhang Li, and Alan Yuille. Searching for trionet: Combining convolution with local and global self-attention. In *BMVC*, 2021a.

Huaijin Pi, Huiyu Wang, Yingwei Li, Zizhang Li, and Alan L. Yuille. Searching for trionet: Combining convolution with local and global self-attention. In *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*, page 141. BMVA

Press, 2021b. URL https://www.bmvc2021-virtualconference.com/assets/papers/0345.pdf.

Pn104. Dataset: 'pneumonia chest x-ray' link. URL https://www.cell.com/cell/fulltext/S0092-8674(18)30154-5.

Nikolay Ponomarenko, Vladimir Lukin, Alexander Zelensky, Karen Egiazarian, Marco Carli, and Federica Battisti. Tid2008-a database for evaluation of full-reference visual quality assessment metrics. *Advances of Modern Radioelectronics*, 10(4):30–45, 2009.

Nikolay Ponomarenko, Lina Jin, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica Battisti, et al. Image database tid2013: Peculiarities, results and perspectives. *Signal processing: Image communication*, 30:57–77, 2015.

PP86. Dataset: 'ppmi' link. URL https://ai.stanford.edu/~bangpeng/ppmi.html#:~:text=People%20Playing%20Musical%20Instrument&text=The%20PPMI%20dataset%20contains%20images,saxophone%2C%20trumpet%2C%20and%20violin.

Vinay Uday Prabhu and Abeba Birhane. Large image datasets: A pyrrhic win for computer vision? In *WACV*, 2021.

Xianbiao Qi, Yu Qiao, Chun-Guang Li, and Jun Guo. Multi-scale joint encoding of local binary patterns for texture and material classification. In *BMVC*, 2013.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Adityanarayanan Radhakrishnan, Charles Durham, Ali Soylemezoglu, and Caroline Uhler. Patchnet: interpretable neural networks for image classification. *Machine Learning for Health (ML4H) Workshop, Neural Information Processing Systems*, 2018.

Tawsifur Rahman, Amith Khandakar, Muhammad Abdul Kadir, Khandaker Rejaul Islam, Khandakar F. Islam, Rashid Mazhar, Tahir Hamid, Mohammad Tariqul Islam, Saad Kashem, Zaid Bin Mahbub, Mohamed Arselene Ayari, and Muhammad E. H. Chowdhury. Reliable tuberculosis detection using chest x-ray with deep learning, segmentation and visualization. *IEEE Access*, 8:191586–191601, 2020. doi: 10.1109/ACCESS.2020.3031384.

Tawsifur Rahman, Amith Khandakar, Yazan Qiblawey, Anas Tahir, Serkan Kiranyaz, Saad Bin Abul Kashem, Mohammad Tariqul Islam, Somaya Al Maadeed, Susu M. Zughaier, Muhammad Salman Khan, and Muhammad E.H. Chowdhury. Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images. *Computers in Biology and Medicine*, 132:104319, 2021. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2021.104319. URL https://www.sciencedirect.com/science/article/pii/S001048252100113X.

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

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017b.

Mark B. Ring. *Continual Learning in Reinforcement Environments*. PhD thesis, University of Texas at Austin, Austin, Texas 78712, 1994.

Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE workshop on applications of computer vision*, pages 138–142. IEEE, 1994.

A Nene Sameer, K Nayar Shree, and Hiroshi Murase. Columbia object image library (coil-20). *Tech. Rep., technical report CUCS-005-96*, 1996a.

A Nene Sameer, K Nayar Shree, and Hiroshi Murase. Columbia object image library (coil-100). *Tech. Rep., technical report CUCS-006-96*, 1996b.

Sc70. Dataset: '15 scenes' link. URL `https://web.archive.org/web/20070829035029/http://www-cvr.ai.uiuc.edu/ponce_grp/data/index.html`.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4535–4544. PMLR, 2018. URL `http://proceedings.mlr.press/v80/schwarz18a.html`.

Se49. Dataset: 'semeion' link. URL `https://archive.ics.uci.edu/ml/datasets/semeion+handwritten+digit`.

Sh98. Dataset: 'shanghaitech' link. URL `https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Zhang_Single-Image_Crowd_Counting_CVPR_2016_paper.pdf`.

Alireza Shafaei, Mark Schmidt, and James J Little. A less biased evaluation of out-of-distribution sample detectors. In *BMVC*, 2019.

Arash Shahriari. Learning of separable filters by stacked fisher convolutional autoencoders. In *BMVC*, 2016.

Lavanya Sharan, Ruth Rosenholtz, and Edward H. Adelson. Accuracy and speed of material categorization in real-world images. *Journal of Vision*, 14(10), 2014.

Viktoriia Sharmanska and Novi Quadrianto. Learning from the mistakes of others: Matching errors in cross-dataset learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3975, 2016.

Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9630–9638, 2021.

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

sk63. Dataset: 'sketch dataset' link. URL https://cybertron.cg.tu-berlin.de/eitz/projects/classifysketch/.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, number CONF. Omnipress, 2010.

Suraj Srinivas and R Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015.

St34. Dataset: 'stanford cars' link. URL http://ai.stanford.edu/~jkrause/cars/car_dataset.html.

St35. Dataset: 'stanford dogs' link. URL http://vision.stanford.edu/aditya86/ImageNetDogs/main.html.

ST77. Dataset: 'stl10' link. URL https://paperswithcode.com/dataset/stl-10.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, (0):–, 2012. ISSN 0893-6080. doi: 10.1016/j.neunet.2012.02.016. URL http://www.sciencedirect.com/science/article/pii/S0893608012000457.

Yu Su, Moray Allan, and Frédéric Jurie. Improving object classification using semantic attributes. In *Proc. BMVC*, pages 26.1–10, 2010. ISBN 1-901725-40-5. doi:10.5244/C.24.26.

SU26. Dataset: 'sun 397' link. URL https://vision.princeton.edu/projects/2010/SUN/.

SU43. Dataset: 'sun attribute' link. URL https://cs.brown.edu/~gmpatter/sunattributes.html.

SV36. Dataset: 'svhn' link. URL http://ufldl.stanford.edu/housenumbers/.

Sy105. Dataset: 'synthetic covid-19 chest x-ray dataset' link. URL https://github.com/hasibzunair/synthetic-covid-cxr-dataset.

Yaniv Taigman, Lior Wolf, and Tal Hassner. Multiple one-shots for utilizing class label information. In *BMVC*, 2009.

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.

Mingxing Tan and Quoc V Le. Mixconv: Mixed depthwise convolutional kernels. In *BMVC*, 2019.

Wei Ren Tan, Chee Seng Chan, Hernan Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2019. doi: 10.1109/TIP.2018.2866698. URL `https://doi.org/10.1109/TIP.2018.2866698`.

Enoch Tetteh, Joseph Viviano, Yoshua Bengio, David Krueger, and Joseph Paul Cohen. Multi-domain balanced sampling improves out-of-distribution generalization of chest x-ray pathology prediction models. In *Medical Imaging meets NeurIPS workshop*, 2021.

C. Thornton, F. Hutter amd H. Hoos, and K. Leyton-Brown. Auto-weka: combined selection and hyperparameter optimization of classification algorithms. In *KDD*, pages 847–855, 2013.

Sebastian Thrun. A lifelong learning perspective for mobile robot control. *Proceedings of the IEEE/RSJ/GI Conference on Intelligent Robots and Systems*, 1994.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

Sebastian Thrun and Lorien Y. Pratt. Learning to learn: Introduction and overview. In Sebastian Thrun and Lorien Y. Pratt, editors, *Learning to Learn*, pages 3–17. Springer, 1998. doi: 10.1007/978-1-4615-5529-2_1. URL `https://doi.org/10.1007/978-1-4615-5529-2_1`.

TI50. Dataset: 'tid2008' link. URL `http://www.ponomarenko.info/tid2008.htm`.

TI51. Dataset: 'tid2013' link. URL `http://www.ponomarenko.info/tid2013.htm`.

Ti78. Dataset: 'tiny imagenet' link. URL `https://paperswithcode.com/dataset/tiny-imagenet`.

Radu Timofte and Luc Van Gool. Sparse representation based projections. In *Proceedings of the 22nd British machine vision conference-BMVC 2011*, pages 61–61. BMVA Press, 2011.

Radu Timofte and Luc Van Gool. A training-free classification framework for textures, writers, and materials. In *BMVC*, volume 13, page 14, 2012.

Radu Timofte, Karel Zimmermann, and Luc Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. In *2009 Workshop on Applications of Computer Vision (WACV)*, pages 1–8, 2009. doi: 10.1109/WACV.2009.5403121.

Radu Timofte, Karel Zimmermann, and Luc Van Gool. Multi-view traffic sign detection, recognition, and 3d localisation. *Machine vision and applications*, 25(3):633–647, 2014.

Antonio Torralba, Kevin P. Murphy, William T. Freeman, and Mark A. Rubin. Context-based vision system for place and object recognition. In *IEEE Intl. Conference on Computer Vision (ICCV)*, 2003.

Tr79. Dataset: 'trancos' link. URL `https://gram.web.uah.es/data/datasets/trancos/index.html`.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgAGAVKPr.

Eleni Triantafillou, Hugo Larochelle, Richard Zemel, and Vincent Dumoulin. Learning a universal template for few-shot dataset generalization. In *International Conference on Machine Learning*, pages 10424–10433. PMLR, 2021.

Tu106. Dataset: 'tubercolosis' link. URL https://ieeexplore.ieee.org/document/9224622.

UI24. Dataset: 'uiuc cars' link. URL http://host.robots.ox.ac.uk/pascal/VOC/databases.html#UIUC.

UI58. Dataset: 'uiuc texture' link. URL https://web.archive.org/web/20070829035029/http://www-cvr.ai.uiuc.edu/ponce_grp/data/index.html.

UM19. Dataset: 'umist' link. URL http://eprints.lincoln.ac.uk/id/eprint/16081/.

UM59. Dataset: 'umd' link. URL http://users.umiacs.umd.edu/~fer/website-texture/texture.htm.

US52. Dataset: 'usps' link. URL https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html#usps.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013. doi: 10.1145/2641190.2641198. URL http://doi.acm.org/10.1145/2641190.264119.

Vladimir Vapnik. *Statistical learning theory*. Wiley New York, 1998.

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant CNNs for digital pathology. June 2018.

Tom Veniat, Ludovic Denoyer, and MarcAurelio Ranzato. Efficient continual learning with modular networks and task-driven priors. In *International Conference on Learning Representations*, 2021.

Yashaswi Verma and CV Jawahar. Exploring svm for image annotation in presence of confusing labels. In *BMVC*, 2013.

Vi54. Dataset: 'vistex' link. URL https://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html.

VO37. Dataset: 'voc actions' link. URL http://host.robots.ox.ac.uk/pascal/VOC/voc2012/.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. On calibration and out-of-domain generalization. *Advances in neural information processing systems*, 34:2215–2227, 2021.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, M Bagheri, and R Summers. Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *IEEE CVPR*, volume 7, 2017.

Y. Wang and s. Gong. Conditional random field for natural scene categorization. In *Proc. BMVC*, pages 59.1–59.10, 2007. ISBN 1-901725-34-0. doi:10.5244/C.21.59.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

Tobias Weyand, Thomas Deselaers, and Hermann Ney. Log-linear mixtures for object class recognition. In *Proc. BMVC*, pages 30.1–30.11, 2009. ISBN 1-901725-39-1. doi:10.5244/C.23.30.

Wi65. Dataset: 'wikipaintings' link. URL `https://github.com/cs-chan/ArtGAN/tree/master/WikiArt%20Dataset`.

Maciej Wolczyk, Michal Zajac, Razvan Pascanu, Lukasz Kucinski, and Piotr Milos. Continual world: A robotic benchmark for continual reinforcement learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 28496–28510. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper/2021/file/ef8446f35513a8d6aa2308357a268a7e-Paper.pdf`.

Ruobing Wu, Yizhou Yu, and Wenping Wang. Scale: Supervised and cascaded laplacian eigenmaps for visual object recognition based on nearest neighbors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 867–874, 2013.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning, 2019.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. 2017.

Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119(1):3–22, 2016.

Yong Xu, Hui Ji, and Cornelia Fermuller. A projective invariant for textures. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1932–1939. IEEE, 2006.

Yong Xu, SiBin Huang, Hui Ji, and Cornelia Fermuller. Combining powerful local and global statistics for texture description. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 573–580. IEEE, 2009a.

Yong Xu, Hui Ji, and Cornelia Fermüller. Viewpoint invariant texture description using fractal analysis. *International Journal of Computer Vision*, 83(1):85–100, 2009b.

Yong Xu, Sibin Huang, Hui Ji, and Cornelia Fermüller. Scale-space texture description on sift-like textons. *Computer Vision and Image Understanding*, 116(9):999–1013, 2012.

Jimei Yang and Ming-Hsuan Yang. Learning hierarchical image representation with sparsity, saliency and locality. In *Proceedings of the British Machine Vision Conference*, pages 19.1–19.11. BMVA Press, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.19.

Tianbao Yang. Deep auc maximization for medical image classification: Challenges and opportunities. In *Medical Imaging meets NeurIPS workshop*, 2021.

Bangpeng Yao and Li Fei-Fei. Grouplet: A structured image representation for recognizing human and object interactions. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 9–16. IEEE, 2010.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. The visual task adaptation benchmark. *CoRR*, abs/1910.04867, 2019. URL `http://arxiv.org/abs/1910.04867`.

Junjian Zhang, Chun-Guang Li, Chong You, Xianbiao Qi, Honggang Zhang, Jun Guo, and Zhouchen Lin. Self-supervised convolutional subspace clustering network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5473–5482, 2019a.

Weihe Zhang, Yali Wang, and Yu Qiao. Metacleaner: Learning to hallucinate clean representations for noisy-labeled visual recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7365–7374, 2019b. doi: 10.1109/CVPR.2019.00755.

Yimeng Zhang and Tsuhan Chen. Weakly supervised object recognition and localization with invariant high order features. In *Proc. BMVC*, pages 47.1–11, 2010. ISBN 1-901725-40-5. doi:10.5244/C.24.47.

Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016.

Rui-Wei Zhao, Jianguo Li, Yurong Chen, Jia-Ming Liu, Yu-Gang Jiang, and Xiangyang Xue. Regional gating neural networks for multi-label image classification. In *BMVC*, pages 1–12, 2016.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

Hasib Zunair and A Ben Hamza. Synthesis of COVID-19 chest X-rays using unpaired image-to-image translation. *Social Network Analysis and Mining*, 11(1):1–12, 2021.