Bad Students Make Great Teachers: Active Learning Accelerates Large-Scale Visual Understanding

Talfan Evans[1,*] Shreya Pathak[1,*] Hamza Merzic[1,2,*]
Jonathan Schwarz[1,†] Ryutaro Tanno[1] Olivier J. Hénaff[1,*]

[1]Google DeepMind [2]University College London

Abstract

We propose a method for accelerating large-scale pretraining with online data selection policies. For the first time, we demonstrate that model-based data selection can reduce the total computation needed to reach the performance of models trained with uniform sampling. The key insight which enables this "compute-positive" regime is that small models provide good proxies for the loss of much larger models, such that computation spent on scoring data can be drastically scaled down but still significantly accelerate training of the learner. These data selection policies also strongly generalize across datasets and tasks, opening an avenue for further amortizing the overhead of data scoring by re-using off-the-shelf models and training sequences. Our methods, ClassAct and ActiveCLIP, require 46% and 51% fewer training updates and up to 25% less total computation when training visual classifiers on JFT and multimodal models on ALIGN, respectively. Finally, our paradigm seamlessly applies to the curation of large-scale image-text datasets, yielding a new state-of-the-art in several multimodal transfer tasks and pretraining regimes.

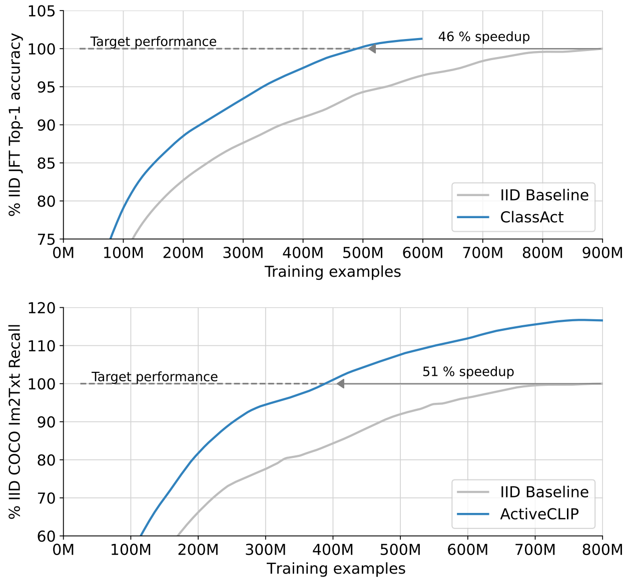

Figure 1. **Active learning accelerates large scale visual understanding**. For large scale classification and multimodal learning tasks, prioritised training on data selected using our active selection methods *ClassAct* (**top**) and *ActiveCLIP* (**bottom**) requires significantly fewer updates than training on data sampled IID.

1. Introduction

In recent years, the explosion of available data has catalyzed the development of increasingly sophisticated models across domains. However, maintaining this progress comes at a cost—power-law scaling for vision and language models [19, 44] dictates that each increment in performance will require another order of magnitude in computation.

One of the key assumptions of these empirical power-laws is that training data is sampled uniformly throughout training. In contrast, active learning approaches sample data in a prioritized fashion, such that computation is spent training on the data that maximally contributes to task performance [22, 25, 35], with the ultimate goals of improving data efficiency and reducing the cost of training runs.

In the first instance, data selection based on hand-engineered features, such as removing images that are the not of a specified shape or that only contain a single colour [2] can trivially improve training efficiency at minimal computational overhead. However, such heuristics are limited in their effectiveness by the expertise of the human designer and are not guaranteed to transfer to training of different models, data modalities or tasks.

In-contrast, recent model-based filtering heuristics have shown promise in small-to-medium visual classification tasks by focusing training on 'hard' and omitting 'easy' data [36], as evaluated by the loss of the learner model itself.

[*]Equal technical contribution. [†]Current affiliation: Harvard University, work done while at Google DeepMind.

Similar gains in data efficiency have been observed in language modeling, where pre-trained models are employed to exclude low-quality data [12]. However, these methods often spend as much computation on the curation of datasets as is gained from subsequent pretraining, with the consequence that training is overall less compute-efficient than training on uniformly-sampled (IID) data.

In this work we investigate whether data-selection policies can instead be designed to be *compute-positive*, such that the sum of all computation needed to reach a target performance, including the pre-training of reference models and inference required for data selection, is less than the computation incurred by an equivalent IID training run.

To address this question, we explore a new model-based active learning framework in which data is scored for prioritised sampling using models that are *significantly smaller and cheaper than the learner model being trained*. The contributions of our work are summarized below:

1. **Benchmarking heuristics for large-scale pretraining:** We conduct an initial investigation of various scoring functions that quantify the importance of examples during training, including loss-based prioritization [13, 17, 20, 36] and 'learnability scores' [28]. From this we find that noise-removal with pretrained models to be an essential component for accelerating learning from large-scale datasets, with efficiency gains up to 46%.
2. **Generalizing selection policies across model scale:** We investigate whether scoring metrics derived from a particular model scale generalize to larger models, and find learnability scores to be uniquely robust, with $1000\times$ smaller scoring models still providing significant efficiency gains. Thanks to these efficient scoring models, we reach the *compute-positive* regime, matching the baseline performance with 25% fewer FLOPs.
3. **Multimodal data curation:** We find our framework to provide a powerful tool for multimodal data curation. Using reference models pretrained on small, clean datasets, we substantially accelerate and improve pretraining on much larger, noisier datasets, yielding strong performance on several multimodal understanding tasks.
4. **Amortizing data selection policies**: We demonstrate that data-selection policies can be trained on one task, but used to accelerate the training of subsequent models on different but related tasks, suggesting that such policies can be easily derived from pre-trained models.
5. **Simplifying active-learning with reference models:** Lastly, we demonstrate that pre-trained reference models may not be necessary at all, where these models are small and can be trained in parallel in an accelerated manner on larger batches than the learner model, while still being compute-positive.

2. Related Work

Data pruning. One approach to data-selection is to identify and sub-select data ahead of training. Paul et al. [29] show that norms of the gradient or prediction error can be used to discard around half of CIFAR10 with no loss in performance. Sorscher et al. [36] go further and argue that data pruning to create 'foundation datasets' can in theory reduce neural scaling laws from power-law to exponential given access to a high-quality scoring metric. These methods have since been deployed for the curation of web-scale datasets in both language modeling [27] and multimodal learning [1, 26], demonstrating large reductions in the amount of data required together with performance improvements. A potential criticism of this line of work is the repeated observation of sequential concept learning even in an IID training run [3, 7, 31], which static data-filtering pipelines cannot accommodate due to their inability to adapt to the state of the model. More critically, in the single-epoch regime that is becoming typical of large model training [14, 19], pre-filtering can be as expensive as learning from it, a shortcoming which we address in this work.

Online active learning. The natural alternative to the 'foundation dataset' approach is to continuously filter data throughout training. Unlike data-pruning, online active learning applies naturally to the semi-infinite / single-epoch regime. One approach (Online Batch Selection [23]) is to score, filter and train all using the same learner model, which has the theoretical advantage that the importance of data can be determined relative to the current state of the learner. In terms of metrics, the Reducible Holdout Loss (RHO) [28] also uses the concept of a reference model to identify learnable data points not yet well represented. Other proposed heuristics include memorization for long-tailed data [10] and assigning "complexity" scores based on the number of times the example is forgotten during training [40]. None of these approaches however have demonstrated that the cost of scoring can be mitigated to the point of justifying learner efficiency gains. Most related to our work is DoReMi [41], which uses a small proxy model in order to determine the optimal data-mixture weights for the subsequent training of a larger model.

Meta-learning. Finally, there have also been combinations of related ideas in the meta-learning literature, which share the goal of improving overall training efficiency at the cost of per-update complexity. An example is Tack et al. [39], who introduce online data selection at each step within a short optimisation horizon and demonstrate both impressive memory savings and improved performance. For Q&A tasks in the LLM space, Hu et al. [15] show how a meta-learned weighting model can learn to mask the contribution of unimportant sub-sequences to the overall loss, improving results.

3. Methods

In this section we discuss the motivation, design and practical implementation of our active learning framework. In Section 3.1, we discuss how to parallelise online data selection alongside training. Section 3.2 examines which statistics could be relevant for large-scale data selection. Section 3.3 analyses the cost of existing AL frameworks and motivates the introduction of our own. Section 3.4 describes the details relating to applying our framework to large-scale classification and multi-modal learning.

3.1. Data selection as prioritized replay

We approach this problem from the reinforcement learning perspective, where data generation ('acting') and learning processes are decoupled.

We apply this framework to standard visual learning tasks in the following way: we sample uniformly from the training set $\boldsymbol{x}_i \sim \mathcal{D}$ and assign a score $s_i = s(\boldsymbol{x}_i|\theta)$ to each data point $\boldsymbol{x}_i$ using model parameters θ. Given a large enough collection of scored examples stored in a memory bank $\mathcal{M} = \{\boldsymbol{x}_i\}_{i\in(0,\dots,M-1)}$, we sample *non-uniformly* according to their scores [32]. Specifically, we sample $\boldsymbol{x}_i \overset{\pi}{\sim} \mathcal{M}$, where the policy π is computed as a softmax over scores:

$$\pi(\boldsymbol{x}_i) = \frac{\exp(s_i(\boldsymbol{x}_i))}{\sum_j \exp(s_j(\boldsymbol{x}_j))} \tag{1}$$

The 'actors' continuously read data, compute scores, and write the scored data to the memory bank in parallel to the learning process. This allows maximum utilization of the hardware accelerators needed for learning by offloading the data processing and scoring tasks to actors which can utilize more cost-effective asynchronous computation.

3.2. Statistics for data selection

We explore a few statistics for model-based prioritization, grouped into two categories.

Example difficulty: given the current state of the learner, an intuitive prioritization scheme might favour 'difficult' examples (as measured by their training loss), while removing 'easy' examples that are trivially classified and which yield small gradients. This loss-based prioritization:

$$s^{\text{hard}}(\boldsymbol{x}_i|\theta) = \ell(\boldsymbol{x}_i|\theta) \tag{2}$$

can use the current parameters of the learner θ^t or those of a fixed model θ^*.

The opposite argument can been made for favoring examples that are *easily* solved by a well-trained model, as such a prioritization removes the noisy examples present in large-scale datasets:

$$s^{\text{easy}}(\boldsymbol{x}_i|\theta) = -\ell(\boldsymbol{x}_i|\theta) \tag{3}$$

This prioritization scheme is commonly used in multimodal learning for identifying high-quality examples with pre-trained models [13, 33, 34].

Example learnability: Given that favoring easy and hard examples target different and potentially orthogonal properties of the data, a natural question is whether these policies can be combined. Learnability criteria straightforwardly combine the two as

$$\begin{aligned} s^{\text{learn}}(\boldsymbol{x}_i|\theta^t,\theta^*) &= s^{\text{hard}}(\boldsymbol{x}_i|\theta^t) + s^{\text{easy}}(\boldsymbol{x}_i|\theta^*) && (4)\\ &= \ell(\boldsymbol{x}_i|\theta^t) - \ell(\boldsymbol{x}_i|\theta^*), && (5)\end{aligned}$$

favoring examples that are easily solved by a well-trained model θ^* but challenging to the learner in its current state θ^t, such that more computation dedicated to this example could lower its loss. Conversely, examples that are trivially classified by the learner (or mislabeled) will yield low (or high) losses for *both* the current learner and the well-trained one, leading to low learnability scores.

A special case of learnability scores (the Reducible Hold-Out Loss [28]) uses a model θ^{ho} specifically trained on a held-out dataset to ensure the independence of its predictions from those of the current learner $s^{\text{learn}}(\boldsymbol{x}_i|\theta^t,\theta^{\text{ho}})$. We assess in Section 4.2 whether this is necessary when training on large-scale image datasets.

3.3. Unlocking compute-positive training

While model-based data-selection policies have demonstrated savings in terms of learner computation, none (to the best of our knowledge) produce overall savings when also accounting for the computation associated with the data selection process, which requires taking forward passes over candidate data. Although the cost of an inference pass F is $\sim$1/3 the cost of a gradient update ($3F$) for most models, this cost scales with the proportion of data which is being rejected (*e.g.* retaining only 20% of the data requires 5 inference passes per trained batch). The requirements for *compute-positivity* can therefore be expressed as:

$$\underbrace{\Big(3F_{\text{learn}} + \rho F_{\text{act}}\Big)\beta + 3F_{\text{ref}}}_{\text{Active Learning}} < \underbrace{3F_{\text{learn}}}_{\text{IID}} \tag{6}$$

where F_{act} is the cost of scoring an example, ρ is the number of examples scored per training example, and β is the saving relative to IID training in terms of learner updates (see Appendix Section A.1 for more details). The RHS term is the cost of IID training per update. The first LHS term inside the brackets is the cost of the learner training during AL, the second term is the scoring cost and the last term outside the brackets the cost of training the reference model. We illustrate in Figure 2 the different contexts in which parts of this computation may be effectively amortized.

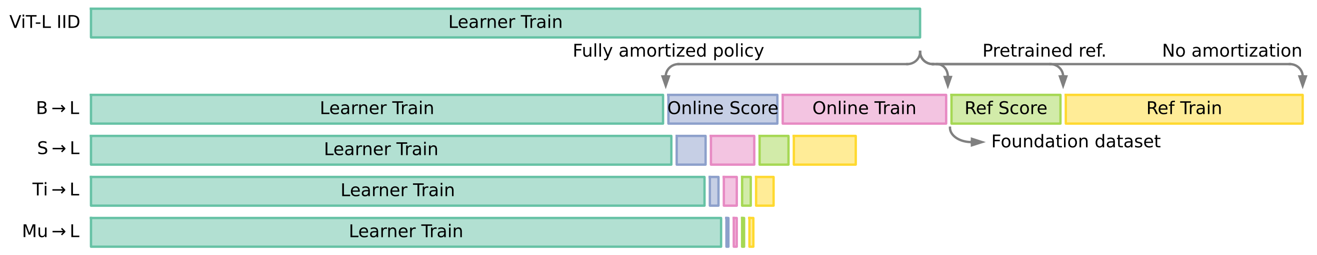

Figure 2. **Amortizing the cost of data selection**. Draw to scale - length of bars proportional to FLOP counts from Figure 4. Expensive model policies (e.g. using a ViT-B to score data for the ViT-L learner, or 'B → L') produce better learner speedups, at the expense of the additional FLOPs associate with data selection. However, some or part of these costs can be amortized, depending on the context. Using off-the-shelf reference models removes the need to train from scratch (yellow). Since the reference model is fixed throughout training, scores can be assigned once to a 'foundation dataset' and amortized across many training runs (lime green; [28, 36]). Since the online model is independent of the learner model and generalizes across scale, data selection policies can also be distilled as a fixed ordering of a given dataset (a 'foundation curriculum'). Even in the case where no amortization is possible, small model policies produce net savings in FLOPs over IID training, at the expense of marginal decreases in speedup in terms of learner FLOPs.

In the streaming and/or large-scale model training regime where data is not repeated nor seen before, in order to be *compute-positive*, either/all of the reference model training, actor scoring and learner efficiency β terms must be made smaller to produce net savings vs. IID. Typical prioritization schemes can produce saving on the order of 50% (*i.e.* $\beta = 0.5$), suggesting that savings must also be made by down-scaling the other terms F_{act} and F_{ref}.

Cost of easy-reference scoring. While both the cost of the reference model and example scoring can be scaled down in the case of easy-reference scoring ($F_{\text{act}} = F_{\text{ref}}$, see equation 3), it is unclear whether the efficiency gains β are robust to this down-scaling (see section 4.2).

Cost of RHO learnability scoring. The original definition of learnability scores [28] requires inference passes through both the learner and a reference model ($F_{\text{act}} = F_{\text{ref}} + F_{\text{learn}}$, see equation 4), meaning that although the cost of the reference model can be scaled down, the cost of example scoring cannot.

Cost of ClassAct / ActiveCLIP. For this reason, we explore whether replacing the learner model in term 1 of equation 4 with a much smaller model can still produce comparable learner efficiency gains to those already observed. Specifically, we introduce a third "online" model, which has the same architecture and size as the reference model, but is trained in parallel with the learner (see Algorithm 1). In this case, the cost of scoring examples reduces to:

$$F_{\text{act}} = F_{\text{ref}} + F_{\text{online}} = 2F_{\text{ref}} \quad (7)$$

and can be arbitrarily scaled down along with the reference model. We instantiate our method for two canonical pre-training tasks: visual classification and multimodal learning, which we call *ClassAct* and *ActiveCLIP* respectively.

Algorithm 1 *ClassAct / ActiveCLIP*

```
1: Input: Randomly initialized learner model $\theta_l$ and small online model $\theta_o$, small pre-trained reference model $\theta_r$. Models use loss $\ell_{\text{act}}$ for scoring data and $\ell_{\text{learn}}$ for computing updates. Dataset $\mathcal{D}$, batch size $B$, sub-batch size $b < B$.
2: while training do
3:     $X \sim \mathcal{D}$, where $|X| = B$                                   ▷ Sample IID
4:     $S = \ell_{\text{act}}(X|\theta_o) - \ell_{\text{act}}(X|\theta_r)$                ▷ Get scores
5:     $I \sim \text{SoftMax}(S)$, where $|I| = b$                   ▷ Sample indices
6:     $Y = X[I]$                                              ▷ Collect sub-batch
7:     $\theta_l \leftarrow \text{Adam}[\nabla_{\theta_l}\ell_{\text{learn}}(Y|\theta_l)]$         ▷ Update learner model
8:     $\theta_o \leftarrow \text{Adam}[\nabla_{\theta_o}\ell_{\text{learn}}(Y|\theta_o)]$         ▷ Update online model
9: end while
```

3.4. Losses for canonical visual pre-training tasks

For visual classification, we use the standard cross-entropy loss for both actors and learners, i.e.

$$\ell_{\text{act}}(\boldsymbol{x}_i, \boldsymbol{y}_i|\theta) = \ell_{\text{learn}}(\boldsymbol{x}_i, \boldsymbol{y}_i|\theta) = \text{XEnt}(\boldsymbol{y}_i, \boldsymbol{l}_i(\theta)), \quad (8)$$

where $\boldsymbol{l}_i(\theta)$ are the logits produced by the model. For multimodal learning, learners optimize the contrastive loss $\ell_{\text{learn}} = \ell^{\text{im,txt}}_{\text{learn}} + \ell^{\text{txt,im}}_{\text{learn}}$, whereas the actor loss ℓ_{act} is simply the dot-product similarity between image and text embeddings:

$$\ell_{\text{act}}(\boldsymbol{x}_i|\theta) = -\boldsymbol{z}^{\text{im}}_i \cdot \boldsymbol{z}^{\text{txt}}_i \quad (9)$$

$$\ell^{\text{im,txt}}_{\text{learn}}(\boldsymbol{x}_i|\theta) = -\log \frac{\exp(\boldsymbol{z}^{\text{im}}_i \cdot \boldsymbol{z}^{\text{txt}}_i)}{\sum_j \exp(\boldsymbol{z}^{\text{im}}_i \cdot \boldsymbol{z}^{\text{txt}}_j)} \quad (10)$$

where $\boldsymbol{z}^{\text{im}}_i = f^{\text{im}}(\boldsymbol{x}_i;\theta)$ and $\boldsymbol{z}^{\text{txt}}_i = f^{\text{txt}}(\boldsymbol{x}_i;\theta)$ are image and text embeddings respectively, and $\ell^{\text{txt,im}}_{\text{learn}}(\boldsymbol{x}_i;\theta)$ is defined analogously to $\ell^{\text{im,txt}}_{\text{learn}}(\boldsymbol{x}_i;\theta)$.

4. Experiments

All our experiments were conducted with Vision Transformers [8] for which strong baselines are available across model sizes [44]. Unless specified, we adopt models with patch size 16 throughout (ViT-S refers to ViT-S/16 and similar). Since we are interested in studying active learning for large-scale pre-training, we consider two canonical regimes: large-scale classification on JFT-300M [37] and multimodal contrastive learning [30]. When pre-training with JFT classification we use held-out classification performance as the evaluation metric. When pre-training on image-text data we evaluate with standard multimodal transfer tasks: ImageNet zero-shot classification and image-to-text / text-to-image retrieval on COCO.

Throughout, we will refer to the large batch of size B sampled IID from the training data as the 'super-batch', and the prioritised smaller batch of size $b < B$ as the 'sub-batch'. In all our experiment, we filter 50% of IID sampled data such that $\rho = B/b = 2$, although more aggressive filtering regimes warrant investigation [36].

Our models were developed using the JAX ecosystem [4, 5] with efficient prioritized replay based on [6].

4.1. Evaluating loss-based scoring heuristics in the large-data regime

We begin by evaluating a suite of loss-based scoring heuristics on their ability to accelerate supervised classification on the JFT dataset. Figure 3 shows the results of training an 86M parameter ViT-B with various sampling strategies. Arguably the most intuitive method to score data is to prioritise training on data with high loss under the learner. This has been shown to be effective in the small data regime [11, 23] but has not been successfully scaled. In our experiment, this *hard learner* prioritisation (equation 2) marginally improved performance at the end of training but was not significantly better than the IID baseline, despite requiring an additional inference pass over the super-batch. This is perhaps not surprising - data points with high loss may also be unlearnable due to *e.g.* label noise, such that training on those data points does not result in the model performing any better on the held out test set. Large scale datasets are more likely to be noisy.

Scoring methods based on pre-trained reference models performed much better—both *easy reference* prioritisation (equation 3), and prioritising based on *learnability* (equation 4) produced significant gains over IID sampling. Here, we pre-trained an identical ViT-B for the same 3 epochs to use as a reference model for a second training run displayed above. In our experiments, *learnability* based scoring slightly underperforms *easy reference* despite requiring an extra inference pass on the learner. Both models produced speed-ups relative to IID training of 33% (Fig. 3).

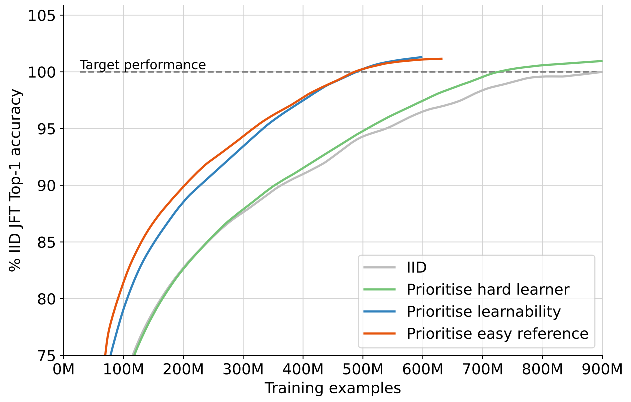

Figure 3. **Evaluation of loss-based data-selection criteria for large-scale classification**. We train a ViT-B on JFT-300M with different data-selection policies. Prioritising hard data under the learner (green curve) produced late and marginal gains over the IID baseline. Prioritizing data using both *learnability* (blue curve, [28]) and ease with respect to a pre-trained reference model (red curve, [13]) produced significant speedups and performance gains.

4.2. Generalising data-selection policies across scale

Although the speed-ups afforded by *learnability* prioritization (Fig. 3) are significant in terms of learner FLOPs, they come at the cost of the additional inference passes required to score the data during learner training, plus the cost of training the reference model. This makes the overall gains strongly *compute-negative* relative to IID training (in addition to requiring extra time for reference model pre-training). Even if the size of the reference model is scaled down [28], these methods still incur the cost of additional learner inference passes to score the data during training.

As discussed (Section 3.3), although *easy-reference* scoring could be amortized by constructing 'foundation datasets', the same savings could not be recovered in the semi-infinite data regime where data may never be repeated, or when training continually larger models.

Unlocking compute-positive active learning. To address this issue, we introduce a set of down-scaled models with the same ViT architecture that we use to score data for training a larger ViT-L model (see Methods Section 3.3). Specifically, we use ViT-B, ViT-S or ViT-Ti variants (which are 4×, 13× and 47× cheaper than the learner model) for both the online and reference model (see Algorithm 1). In Figure 4 (left) we assess the impact of these cheaper scoring models on learner efficiency. First, we find that loss-based prioritization (*i.e. easy reference*) to be very sensitive to the capacity of the scoring model: while ViT-B scoring models yield reasonable gains over IID sampling, prioritizing with ViT-S and ViT-Ti scoring models underperforms significantly (Fig. 4, red curves).

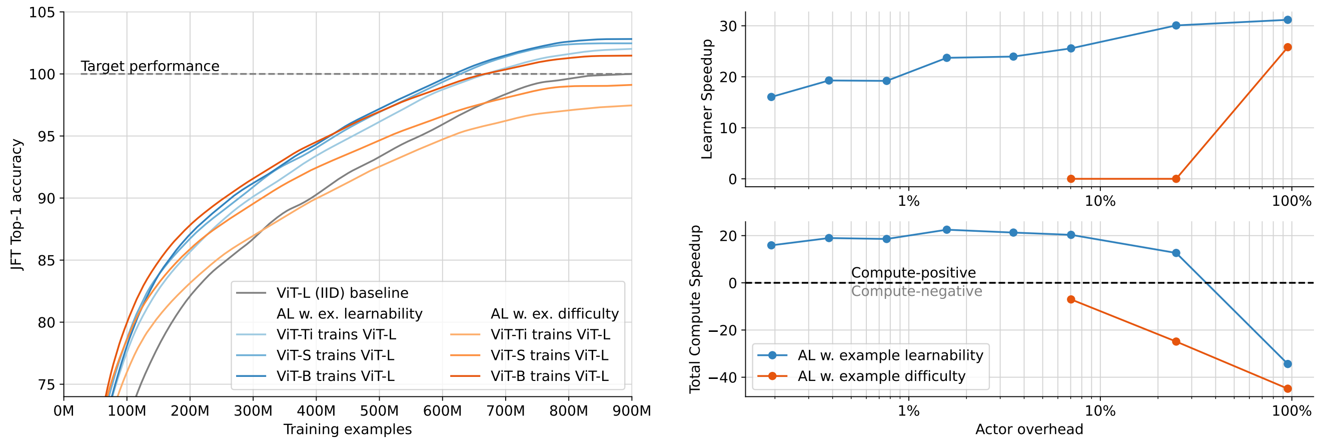

Figure 4. **Generalization of data-selection policies across models scales. Left:** We train a ViT-L for 3 epochs on JFT using IID sampling (grey) or prioritized data sampling using example learnability (blue) or example difficulty (red). Example scores are computed using ViT-B actors (dark), or cheaper ViT-S or ViT-Tiny models (light). While both example learnability and difficulty yield good speedups with expensive actors, learnability criteria are much more robust to approximate scoring. **Top right:** Learner speedup is computed as the fraction of learner iterations saved in order to attain the baseline's top performance. Actor overhead is computed as the the additional computation in FLOPs required to score examples with a particular actor architecture. Example learnability yields robust learner speedups across actor scales, example difficulty does not. **Lower right:** total compute efficiency is calculated as a product of learner efficiency and actor overhead, indicating the amount of computation required to reach the baseline's performance. Approximate actors (*i.e.* ViT-S or smaller) computing example learnability enable total compute speedups, other schemes do not.

In contrast, we find prioritized sampling according to example *learnability* to yield robust gains, even when the scoring models are significantly scaled down (Fig. 4, blue curves). For example, while ViT-B scoring models yield a 31% learner speedup, the 50× smaller ViT-Ti scoring models still provide a 26% speedup. We pushed this logic by using even smaller scoring models (the ViT-Mu family which we introduce, see Appendix A.7) which are up to 1000× smaller than the learner. Despite this, prioritizing data based on their scores yields non-negligeable speedups (*e.g.* 16% for the smallest actors we consider; Figure 4, top right).

These experiments demonstrate that, with the appropriate scoring criterion, online and reference models can be significantly downscaled and still produce comparative gains to the larger models, with learner efficiency degrading gracefully with the actor overhead (*i.e.* the cost of the reference model and data scoring). As a result, our method *ClassAct* quickly becomes FLOP positive as the online + reference models are downscaled (Figure 4, bottom right), while at the same time producing speed-ups in wall-clock time for a given learner batch size.

Together, our results expose a pareto front across which to determine an optimal context-specific data selection strategy (Figure 2). Where pre-trained models are available, some of the cost of larger data selection policies can be discounted. If savings in wall-clock time supersede the associated cost of scoring, large models can be tolerated for data selection. Reference model costs can also be amortized across many training runs by appending scores to 'foundation datasets'. However, in the case where no component of the framework can amortized (as in the case of large-scale pretraining), prioritizing data with small *ClassAct* models can deliver large savings in total computation.

4.3. Generalising neural scaling laws to the active-learning setting

We next investigated the scaling behaviour of *ClassAct* by experimenting over large learner compute budgets (Fig. 5), using both ViT-Ti and ViT-S models as actors for training a ViT-L. Predictably, the ViT-S produced larger, although marginal gains over the ViT-Ti actors when not accounting for scoring FLOPs (Figure 5, left). However, when accounting for total FLOPs, the difference was less pronounced (Figure 5, right). Our results generalize large scale IID scaling laws such as uncovered by [19] for LLMs and reproduced for large vision transformers by [44] to the case of non IID sampling. For the first time, we demonstrate that these scaling laws can be shifted in our favour by selecting data using general model-based scoring heuristics.

4.4. Training the reference model in parallel

The reference model needs to be trained if none is already available. This two-step process adds complexity to active model training, especially if using large scale infrastructure. However, an interesting consequence of down-scaling the reference model is that both inference passes and gradients can be computed over a much larger batch than can be computed on the learner. In theory, this would mean that the

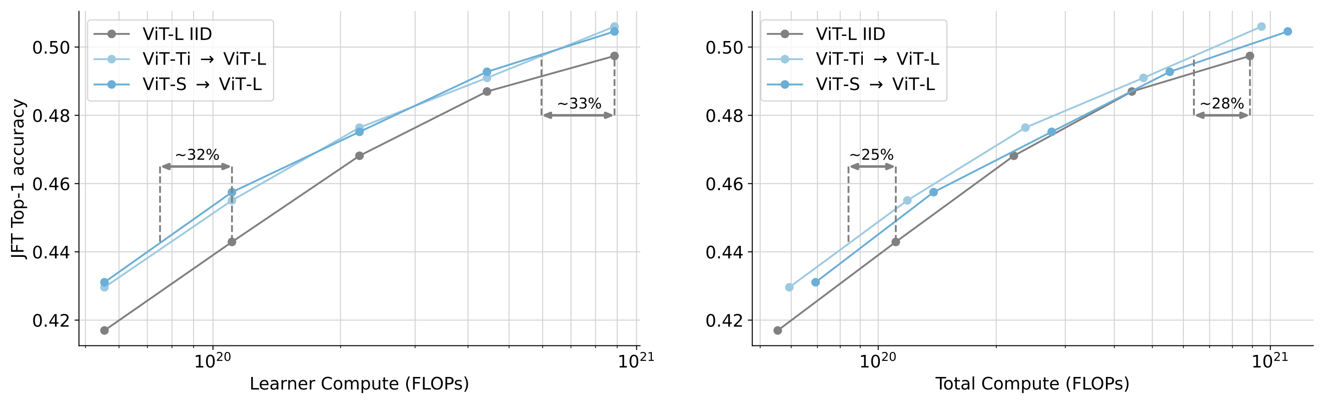

Figure 5. **Scaling laws for active learning**. To test the extent that smaller models can be used to accelerate pre-training for larger models, we trained a baseline ViT-L over a range of compute budgets (this is the compute optimal model size, see Zhai et al., 2021). We also trained the same ViT-L with both ViT-Ti and ViT-S reference policies, pre-trained for the same number of epochs. **Left**: Small model policies produce robust savings in learner compute. **Right**: When accounting for total compute (learner + actor training and data scoring), small model policies in all compute budgets produce FLOP savings over IID training. These scaling laws generalize those measured empirically in the IID setting [44] to the case of non-IID data selection.

	ViT model capacity				Speed-up	
Method	Reference	Online	Learner	Reference Type	Learner speedup	Compute speedup
ViT-B IID			B		0%	0%
RHO [28]	Tiny	B	B	Held-out, fixed	0%	- 79%
ClassAct-HO	Tiny	Tiny	B	Held-out, fixed	18%	3%
ClassAct	Tiny	Tiny	B	In-domain, fixed	18%	3%
ClassAct-Online	Tiny	Tiny	B	Trained online	17%	2%

Table 1. **Simplifying and accelerating the computation of learnability scores.** Relative to RHO *ClassAct* makes two changes: replacing the reference model with one trained in-domain (removing the need for bespoke held-out sets), and dramatically reducing capacity of the online actor models used for scoring examples. All experiments were conducted on 3 epochs of one-half of the JFT dataset to enable the held-out ablations. RHO with a small reference model did not produce a learner speedup in our experiments.

small reference model could instead be trained *online*, in parallel with the large learner and small online model.

We confirmed our hypothesis by running an experiment in which we trained our reference model on a super-batch of size $10b$ and trained the online and learner model in sequence with the sub-batch of size b. To make sure the reference model quickly converges, we additionally set the learning rate to double that of the online and learner models (this would cause instability for the learner and online models because of the additional variance from the smaller batch). We also verified that training the reference model on a held-out set of data performed equally in our experiments to reference models trained on the same data as the learner model [28]. Our 'one-pass' setup, *Online-ClassAct*, produces the same performance as the pre-trained *ClassAct* pipeline in our Ti-trains-B experiments (Table 1). Pseudocode is shown in appendix Algorithm 2.

We have shown that by *decoupling the scoring models from the learner model entirely*, it is possible to significantly downscale the scoring models with minor degradation to performance (see Table 1). Unlike RHO, which can train a large 'learner' model with a small 'reference' model, we introduce a third 'online' model, with the same architecture and parameter count as the reference model, enabling the reduction of actor computation (see Table 1).

4.5. ActiveCLIP: active multimodal learning

We have so far demonstrated that large scale image classifiers can be trained with lower total compute by actively selecting the data used for training. However, classification has largely been superseded as a large scale pre-training method by CLIP-style multimodal training [30]. Figure 6 demonstrates that our CLIP-adapted active learning method *ActiveCLIP* (see Methods) produces similar speedups in terms of learner computation as observed in JFT classification. Specifically, we find that prioritized sampling with learnability scores accelerates multimodal pre-training by 18-48%, depending on the evaluation metric (ImageNet zero-shot accuracy or COCO retrieval) and reference model configuration, which we explore below.

	Reference	Learner (+online)	ImageNet 0-Shot Speed-up	ImageNet 0-Shot Top 1 Accuracy	COCO (im2txt) Speed-up	COCO (im2txt) R@1	COCO (txt2im) Speed-up	COCO (txt2im) R@1
IID		ALIGN	0.0%	45.6	0.0%	38.5	0.0%	25.5
ActiveCLIP	ALIGN	ALIGN	23%	47.8	28.6%	40.9	26.8%	27.3
ActiveCLIP	LTIP	ALIGN	48%	53.2	48.0%	44.8	27.0%	30.5
IID		LTIP	0.0%	46.6	0.0%	44.4	0.0%	30.1
ActiveCLIP	ALIGN	LTIP	0.0 %	46.5	18.8%	45.6	28.2%	31.8
ActiveCLIP	LTIP	LTIP	16%	47.2	22.1%	45.4	17.5%	31.0

Table 2. **Generalizing data-selection policies across datasets and tasks.** We pretrain reference models on the large, noisy ALIGN dataset, or the smaller and more curated LTIP dataset [2]. Consistently with [2], we find IID training on LTIP to yield stronger transfer learning results than pre-training on ALIGN. Interestingly, these reference models can be used very effectively for data-selection on both LTIP and ALIGN, whereas ALIGN-pretrained reference models yield more modest speedups. All models are provided with 800M training images at resolution 128×128, speedups are shown relative to the time at which the IID baseline was reached for that evaluation metric.

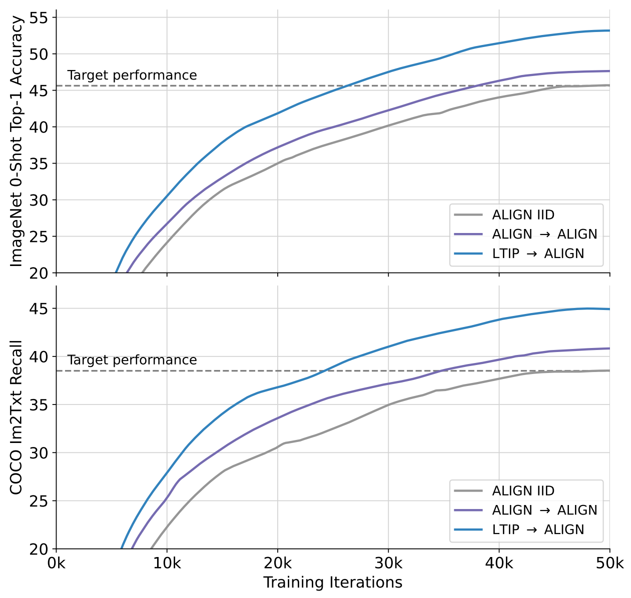

Figure 6. **Reference policies generalize across tasks**. We train a ViT-B reference policy on either ALIGN or LTIP ([2]), then use it to train a second ViT-B learner again on either ALIGN or LTIP. As before, *ActiveCLIP* produces efficiency gains over IID in terms of learner updates. The biggest gains, which surpassed the in-domain *ActiveCLIP* results, were instead found in the out-of-domain setting where a LTIP reference model is used to accelerate an ALIGN training run. See Table 2 for more details.

4.6. Policy generalization across tasks

To fully leverage off-the-shelf pre-trained image models for training (Figure 2), our results up to now suggest that the reference model should be trained on the same task as the learner model is being trained for. In Figure 6 (see Table 2 for full results), we show that we can in fact pre-train our reference models on related but distinct datasets. Moreover results suggest that there may even be a benefit to cross-training in some-cases. Models trained on ALIGN with an ALIGN trained reference model ('in-domain' *ActiveCLIP*) produced similar speedups over models trained IID on ALIGN on downstream metrics, but these gains were greatly surpassed by instead training on the LTIP dataset. One possibility is that LTIP is less noisily labeled such that the scoring policies it produces are 'cleaner' - i.e. more able to filter for 'clean' data. The corollary may also be true; active data selection appears to greatly improve the utility of ALIGN, suggesting that it contains a large proportion of data that is of low quality for training.

Method	Train ex.	IN-1K ZS Top-1	COCO im2txt	COCO txt2im
CLIP [30]	13B	68.3	52.4	33.1
EVA-CLIP [38]	3B+2B	69.7[†]		
ActiveCLIP	3B	**71.3**	**57.7**	**43.0**
OpenCLIP [16]	34B	70.2	59.4	42.3
EVA-CLIP [38]	8B+2B	74.7[†]	58.7	42.2
ActiveCLIP	8B	**72.2**	**60.7**	**44.9**

Table 3. **Comparison of *ActiveCLIP* to public multimodal pre-taining methods**. All models use ViT-B/16 encoders, with 224×224 image resolution, except SigLIP which uses 256×256. ActiveCLIP outperforms models trained with the same or more training data, on ImageNet zero-shot classification and both forms of COCO retrieval. [†]benefits from additional ImageNet21K pre-training (with 2B training examples).

4.7. Comparison to prior multimodal art

We leverage the insight that pre-training a reference model on clean data can facilitate learning on larger, noisy data for training our final model. Here, we train a reference model on LTIP, then use it to train a new model on the much larger mixture of LTIP and ALIGN, following [2], for a total of 3 and 8 billion training examples. Table 3 shows that in this training regime, *ActiveCLIP* surpasses models trained with significantly more data on ImageNet 0-Shot classification and COCO retrieval.

5. Discussion

In this work, we have presented a new method for active data selection that builds upon and simplifies the concept of 'learnability'. Our experiments demonstrate that this approach can significantly reduce the computation required for large-scale pretraining, compared to uniform IID training. To our knowledge, this is the first active learning method that is more efficient than IID training when accounting for total FLOPs, and that does not rely on hand-designed features, allowing broad application across training setups. We have validated this by showing results on classification and contrastive pre-training, and found that our data selection policies continue to produce efficiency gains in the large-scale regime and can generalize effectively across task modalities. Collectively, our experiments also illustrate a Pareto frontier that allows trading off actor/data-selection computation against savings in training iterations, suggesting an alternative path to improved performance beyond scaling training batch sizes.

In this work we focused on supervised pre-training for images, but further work could involve extending our method to other modalities and training schemes such as language, video, and generative modeling. An important note is that all our experiments present results from filtering only 50% of the data; further gains may be possible by filtering more aggressively, at the cost of further overheads. In particular, aggressive data-selection coupled with efficient scoring schemes such as the ones proposed here could test the hypothesis that large-scale pretraining can benefit from exponential, rather than power-law, scaling behavior.

References

[1] Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S Morcos. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*, 2023. 2

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: A visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 2022. 1, 8

[3] Devansh Arpit, Stanisław Jastrzkebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017. 2

[4] Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, et al. The deepmind jax ecosystem, 2020. *URL http://github. com/deepmind*, 2010. 5

[5] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 5

[6] Albin Cassirer, Gabriel Barth-Maron, Eugene Brevdo, Sabela Ramos, Toby Boyd, Thibault Sottiaux, and Manuel Kroiss. Reverb: A framework for experience replay, 2021. 5, 2

[7] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active bias: Training more accurate neural networks by emphasizing high variance samples. *Advances in Neural Information Processing Systems*, 30, 2017. 2

[8] 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. In *International Conference on Learning Representations*, 2021. 5, 1

[9] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR, 2018. 2

[10] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020. 2

[11] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pages 1311–1320. Pmlr, 2017. 5

[12] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023. 2

[13] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 2, 3, 5

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, 2022. 2

[15] Nathan Hu, Eric Mitchell, Christopher D Manning, and Chelsea Finn. Meta-learning online adaptation of language models. *arXiv preprint arXiv:2305.15076*, 2023. 2

[16] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. If you use this software, please cite it as below. 8

[17] Angela H Jiang, Daniel L-K Wong, Giulio Zhou, David G Andersen, Jeffrey Dean, Gregory R Ganger, Gauri Joshi, Michael Kaminksy, Michael Kozuch, Zachary C Lipton,

et al. Accelerating deep learning by focusing on the biggest losers. *arXiv preprint arXiv:1910.00762*, 2019. 2

[18] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017. 1

[19] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 1, 2, 6

[20] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR, 2018. 2

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Eur. Conf. Comput. Vis.*, pages 740–755. Springer, 2014. 1

[22] Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27 (4):986–1005, 1956. 1

[23] Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015. 2, 5

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 1

[25] David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992. 1

[26] Anas Mahmoud, Mostafa Elhoushi, Amro Abbas, Yu Yang, Newsha Ardalani, Hugh Leather, and Ari Morcos. Sieve: Multimodal dataset pruning using image captioning models. *arXiv preprint arXiv:2310.02110*, 2023. 2

[27] Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. When less is more: Investigating data pruning for pretraining llms at scale. *arXiv preprint arXiv:2309.04564*, 2023. 2

[28] Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pages 15630–15649. PMLR, 2022. 2, 3, 4, 5, 7

[29] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34:20596–20607, 2021. 2

[30] 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*, 2021. 5, 7, 8, 1

[31] David Raposo, Adam Santoro, David Barrett, Razvan Pascanu, Timothy Lillicrap, and Peter Battaglia. Discovering objects and their relations from entangled scene representations. *arXiv preprint arXiv:1702.05068*, 2017. 2

[32] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015. 3

[33] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 3

[34] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 3

[35] Burr Settles. Active learning literature survey. 2009. 1

[36] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022. 1, 2, 4, 5

[37] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 5

[38] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023. 8

[39] Jihoon Tack, Subin Kim, Sihyun Yu, Jaeho Lee, Jinwoo Shin, and Jonathan Richard Schwarz. Efficient meta-learning via error-based context pruning for implicit neural representations. *arXiv preprint arXiv:2302.00617*, 2023. 2

[40] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018. 2

[41] Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining, 2023. 2

[42] Fan Yang, Gabriel Barth-Maron, Piotr Stańczyk, Matthew Hoffman, Siqi Liu, Manuel Kroiss, Aedan Pope, and Alban Rrustemi. Launchpad: A programming model for distributed machine learning research. *arXiv preprint arXiv:2106.04516*, 2021. 2

[43] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive captioners are image-text foundation models. In *Transactions on Machine Learning Research*, 2022. 1

[44] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022. 1, 5, 6, 7, 2

A. Appendix: supplementary methods

A.1. Details of total compute calculations

The average FLOPs per learner update for an IID training run is given by:

$$C_{\text{IID}} = 3F_{\text{learn}}$$

where F_{learn} is the cost of a single learner inference pass and a gradient update costs $\sim 3\times$ inference passes [18]. The average FLOPs per learner update for easy-reference prioritisation is:

$$C_{\text{easy\_ref}} = \big(3F_{\text{learn}} + \rho F_{\text{ref}}\big)\beta + 3F_{\text{ref}}$$

where $\rho = B/b$ (set to $\rho = 2.0$ for our experiments) and β is the learner speedup vs. IID. The average FLOPs per learner update for RHO is:

$$C_{\text{RHO}} = \big(3F_{\text{learn}} + \rho(F_{\text{learn}} + F_{\text{ref}})\big)\beta + 3F_{\text{ref}}$$

And for *ActiveCLIP / ClassAct*:

$$C_{\text{ClassAct / ActiveCLIP}} = \big(3F_{\text{learn}} + 2\rho F_{\text{ref}}\big)\beta + 3F_{\text{ref}}$$

The conditions for compute positivity are therefore:

$$\Big(3F_{\text{learn}} + \rho F_{\text{act}}\Big)\beta + 3F_{\text{ref}} < 3F_{\text{learn}}$$

where F_{act} is one of F_{ref}, $\big(F_{\text{learn}} + F_{\text{ref}}\big)$ or $2F_{\text{ref}}$ for "easy reference", RHO or ClassAct / ActiveCLIP, respectively.

A.2. Implementation details: ClassAct

Dataset: We pretrain on the JFT-300M dataset which contains labels for 18,292 classes collected in a semi-automated fashion. JFT labels are therefore noisy (consistently with other web-scale datasets), with approximately 20% of the examples being mislabeled. In the "standard" setting both reference models and subsequent ClassAct models are trained on the full JFT-300M dataset.

Architectures: We use standard vision transformers trained with 224×224 image resolution and a patch size of 16×16. We consider the standard ViT-Base and ViT-Large variants as learners in our in our main study, and smaller-scale variants (ViT-S and ViT-Tiny) when reducing the scoring-model computation. We introduce even smaller variants to scale down actor computation further, and detail the configurations of this "ViT-Micro" family in section A.7.

Learner Training: We train visual classifiers with softmax cross-entropy and label smoothing of 0.1. We use the AdamW optimizer [24] with a cosine learning-rate decay and warmup period of 10k iterations. The maximum learning rate is 0.001 and the weight decay is 10^{-3}.

Evaluation: We evaluate the performance of the learned model by applying it to a held-out set of JFT images, and measuring top-1 accuracy. Previous work has found that in-domain performance on JFT is highly predictive of out-of-distribution generalization [44], hence we leave the study of transfer performance to the multimodal setting (Sec. A.3).

A.3. Implementation details: ActiveCLIP

Datasets: We use paired image-text datasets for contrastive pretraining. We experiment with ALIGN, composed of 1.8B image-text pairs, LTIP, introduced in [2] and consisting of 312M higher-quality images with longer descriptions, and JFT-300M which despite being a classification dataset can be used for contrastive learning by using the labels as the paired text for each image [43].

Architectures: The model consists of a vision and text encoder in the standard contrastive learning setup introduced in [30]. We a vision transformer as the visual encoder and a text transformer encoder. We use the Base variant for both the vision and text transformers on the learner, following [8]. The smaller variants used by scoring models are the same as in the classification setup described above.

Learner Training: As described in Section 3.4 we optimize the model using the standard contrastive loss. We train the entire model with the AdamW optimiser and decay the learning rate using a cosine schedule (following a linear warmup period) with a maximum value of 10^{-3}. We also use a learnable temperature parameter as described in [30] and clip the gradients to a maximum global norm of 1.0. We use a batch size of 16,384 on the learner for each dataset.

Evaluation: We evaluate the model's zero-shot transfer results on standard multimodal benchmarks—image-text retrieval on COCO [21] and zero-shot image classification on ImageNet. For COCO, we use the standard technique of comparing image and text representations across the dataset to find the most similar text (or image) for each image (or text). For ImageNet, we precompute the representations for all predefined labels and calculate the most similar label for each image. For zero-shot classification, we also apply the prompt engineering described in [30] using the released prompts[*]. We average the embeddings for each label across all prompts before calculating similarity with the images.

A.4. Online ClassAct / ClassCLIP

ClassAct / ClassCLIP can also be run in a single training loop, by training the reference model on the super-batch:

[*] https://github.com/openai/CLIP/blob/main/notebooks/Prompt_Engineering_for_ImageNet.ipynb

Algorithm 2 Online ClassAct / ClassCLIP

```
1:  Input: Randomly initialized learner model $\theta_l$, small online and reference models $\theta_o$ and $\theta_r$. Models use loss $\ell_{\text{act}}$ for scoring data and $\ell_{\text{learn}}$ for computing updates. Dataset $\mathcal{D}$, batch size $B$, sub-batch size $b < B$. Fast and slow learning rates $\alpha$ and $\beta$.
2:  while training do
3:      $X \sim \mathcal{D}$, where $|X| = B$                              ▷ Sample IID
4:      $S = \ell_{\text{act}}(X; \theta_o) - \ell_{\text{act}}(X; \theta_r)$            ▷ Get scores
5:      $\theta_r \leftarrow \text{Adam}_\alpha[\nabla_{\theta_r} \ell_{\text{learn}}(X|\theta_l)]$       ▷ Update ref. model
6:      $I \sim \text{SoftMax}(S)$, where $|I| = b$                 ▷ Sample indices
7:      $Y = X[I]$                                          ▷ Collect sub-batch
8:      $\theta_l \leftarrow \text{Adam}_\beta[\nabla_{\theta_l} \ell_{\text{learn}}(Y|\theta_l)]$       ▷ Update learner model
9:      $\theta_o \leftarrow \text{Adam}_\beta[\nabla_{\theta_o} \ell_{\text{learn}}(Y|\theta_o)]$       ▷ Update online model
10: end while
```

We found that $B = 10b$ was sufficient to reproduce the pre-trained reference (Algorithm 2) case where $B = 2b$, as described Section 4.4. It is possible that the super-batch size could be shrunk, but we did not evaluate our method in between these values.

A.5. Analyzing learnability scores

Here we examine learnability scores in the case of small differences between the online and reference models θ^t, θ^*. In this case, we can apply a Taylor expansion of the reference-model loss:

$$\ell(\boldsymbol{x}_i|\theta^*) \approx \ell(\boldsymbol{x}_i|\theta^t) + (\theta^* - \theta^t) \cdot \nabla_\theta \ell(\boldsymbol{x}_i|\theta^t) \quad (11)$$

and learnability scores simplify to the alignment between the gradient of the loss with respect to the online parameters, and the difference between online and reference parameters:

$$s^{\text{learn}}(\boldsymbol{x}_i|\theta^t, \theta^*) = \ell(\boldsymbol{x}_i|\theta^t) - \ell(\boldsymbol{x}_i|\theta^*) \quad (12)$$

$$\approx (\theta^t - \theta^*) \cdot \nabla_\theta \ell(\boldsymbol{x}_i|\theta^t). \quad (13)$$

This yields a simple explanation for the relevance of learnability scores: data points whose gradient aligns well with the "direction of travel" (*i.e.* the difference between the current model state and the fully-trained one) will be prioritized. We can further simplify this expression in the case where the reference model is the result of a single, global update with respect to gradients from a batch of data $\{\boldsymbol{x}_j\}$:

$$\theta^* = \theta^t - \lambda \nabla_\theta \ell(\{\boldsymbol{x}_j\}|\theta) \quad (14)$$

$$s^{\text{learn}}(\boldsymbol{x}_i|\theta^t, \theta^*) \approx \lambda \nabla_\theta \ell(\boldsymbol{x}_i|\theta^t) \cdot \nabla_\theta \ell(\{\boldsymbol{x}_j\}|\theta^t) \quad (15)$$

In this case, the learnability of an example reduces to the alignment of its gradient to those of the batch. In particular, these scores will de-prioritize examples which are trivially solved (small gradients) or noisy (mis-aligned with the batch gradient).

Finally, we note that learnability scores reduce to gradient-norm prioritization [29] if we make the further (more stringent) assumption that the reference model is the result of a single update with respect to *this example's gradient only*:

$$\theta^* = \theta^t - \lambda \nabla_\theta \ell(\boldsymbol{x}_i|\theta) \quad (16)$$

$$s^{\text{learn}}(\boldsymbol{x}_i|\theta^t, \theta^*) \approx s^{\text{grad}}(\boldsymbol{x}_i|\theta^t) = \lambda \left|\left|\nabla_\theta \ell(\boldsymbol{x}_i|\theta^t)\right|\right|^2 \quad (17)$$

While this prioritization effectively discards examples with small gradients, it does not benefit from the denoising properties of the more general formulation of learnability, as noisy examples can exhibit large gradients.

A.6. Learning infrastructure

Our learning infrastructure Figure 7 draws inspiration from distributed reinforcement learning (DRL) [9]. In contrast to DRL, where run loops generate data through interactions with an environment, run loops in our system are modified to read offline data. In addition to obtaining data, run loops run inference on the data via remote inference servers and write the inference outputs to prioritized replay data stores [6]. The distributed nodes of our learning setup are configured and connected using [42].

Data stored is sampled from the prioritized replay based on priorities determined during inference. Remote inference servers continuously run inference at specific batch sizes, in parallel to learning. Parameters of the inference servers are updated after each learning step.

For stability, we control the ratio of data sampled to data inserted in all of our experiments. For example, a samples-per-insert (SPI) ratio of 0.5 means that half of the inserted data does not get sampled, ensuring the learner focuses on the most relevant data. Each data item is sampled at most once. These two constraints impose minimum inference requirements to fully saturate the learner, which we separately tune for each experiment by scaling up the topology of the inference server.

Our infrastructure is designed to work with multiple datasets. In this case, we use separate remote inference server, run loops, and prioritized replay for each dataset, and set the batch size per dataset.

To continuously assess the performance of our models, we add data evaluator nodes which measure performance on held-out datasets while periodically pulling the latest parameters from the learner node.

A.7. Small ViT models

We extended the suite of ViT models presented in [44] by further scaling down the ViT-Ti model as shown in Table 4.

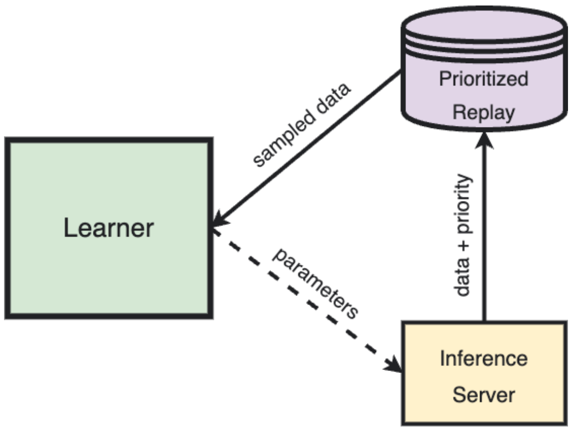

Figure 7. The main nodes of our distributed learning infrastructure are: (1) a learner that continuously updates model parameters based on sampled data, (2) an inference server which computes priorities of uniformly sampled data, and (3) a prioritized replay which receives data and priorities from the inference server and samples prioritized data for learning. The critical aspect of our system is adaptive data prioritization which is enabled by keeping inference parameters in sync with the learner. We employ separate prioritized replay and inference server nodes per dataset. We also employ run loop nodes (not shown in the diagram) for reading and writing data. Run loop nodes can run as threads on the inference server node or remotely, to prevent extra load on inference servers.

Name	Width	Depth	MLP	Heads	Mio- -Params	GFLOPs 224^2
L	1024	24	4096	16	304	61.6
B	768	12	3072	12	86	17.6
S	384	12	3072	12	22	4.6
Ti	192	12	768	3	5.6	1.3
Mu_6-3	192	6	768	3	2.8	1.24
Mu_12-2	128	12	512	2	2.5	1.23
Mu_6-2	128	6	512	2	0.66	0.48
Mu_12-1	64	12	256	1	1.3	0.23
Mu_6-1	64	6	256	1	0.36	0.203
Mu_3-1	64	3	256	1	0.22	0.065
Mu_2-1	64	2	256	1	0.16	0.033
Mu_1-1	64	1	256	1	0.12	0.011

Table 4. **ViT model details**. Configurations, parameter counts, and FLOPs associated with small ViT models.