Adaptive Training Distributions with Scalable Online Bilevel Optimization

David Grangier, Pierre Ablin, Awni Hannun *{grangier, p_ablin, awni}@apple.com*
Apple

Reviewed on OpenReview: *https://openreview.net/forum?id=JP1GVyF5i5*

Abstract

Large neural networks pretrained on web-scale corpora are central to modern machine learning. In this paradigm, the distribution of the large, heterogeneous pretraining data rarely matches that of the application domain. This work considers modifying the pretraining distribution in the case where one has a small sample of data reflecting the targeted test conditions. We propose an algorithm motivated by a recent formulation of this setting as an online, bilevel optimization problem. With scalability in mind, our algorithm prioritizes computing gradients at training points which are likely to most improve the loss on the targeted distribution. Empirically, we show that in some cases this approach is beneficial over existing strategies from the domain adaptation literature but may not succeed in other cases. We propose a simple test to evaluate when our approach can be expected to work well and point towards further research to address current limitations.

1 Introduction

Large models pretrained on massive, heterogeneous datasets have impacted various application domains (Bommasani et al., 2021), including natural language processing (Devlin et al., 2019), computer vision (Mahajan et al., 2018), and audio processing (Schneider et al., 2019). These models are typically trained on two different distributions: a *generic distribution* for pretraining and a *specific distribution* for fine tuning. Only the specific distribution matches the test conditions while the generic distribution offers an abundant source of data with some similarities to the specific data. This novel paradigm builds upon earlier work in multitask learning (Caruana, 1997), transfer learning (Bennett et al., 2003), and domain adaptation (Moore & Lewis, 2010). For all of these methods, the accuracy of a model on the specific task heavily depends on selecting an appropriate distribution over the generic auxiliary tasks and data.

This work proposes a scalable online strategy for data selection along with a comprehensive and realistic empirical study[1]. We build upon a bilevel formulation of the generic re-weighting problem which allows for gradient-based optimization (Franceschi et al., 2018).

Contributions First, we unify several gradient-based data selection methods into a common framework in which their similarities and distinctions are more easily understood. Second, we introduce a scalable, online algorithm. This algorithm can train a large model while updating an inexpensive auxiliary data selection model which tracks the distribution required to make fast progress on the targeted task. Our algorithm leverages the asymmetry in computational cost between the selection model and the large model by filtering examples on the fly, ensuring that the majority of examples are not examined by the large model. This allows for much faster training on the specific distribution than pre-training on the generic distribution only.

Third, we perform a comprehensive and realistic empirical comparison of data selection strategies. We compare several alternative strategies across different tasks and modalities including large scale language

[1]Code to reproduce our experiments is at `https://github.com/apple/ml-bilevel-train-dist`

modeling, machine translation, and image classification. Finally, we propose a simple metric based on gradient alignment that correlates with the success and failure of gradient-based data selection methods.

2 Related Work

Prior work has proposed automatic methods to adjust the generic training distribution in order to improve model generalization on the specific task. The domain adaptation literature has explored variants of importance sampling, which uses importance weights to emphasize or select some generic examples. These weights have been determined via domain classifiers (Aharoni & Goldberg, 2020; Gururangan et al., 2020), via the estimation of the label distribution (Ngiam et al., 2018), or via gradient alignment and fine-tuning.

Contrastive Data Selection, CDS (Moore & Lewis, 2010; van der Wees et al., 2017; Wang et al., 2018) falls into this later category. This method has four phases: (i) an initial model is pre-trained on the generic dataset, (ii) this model is fine tuned on the specific data, (iii) the generic set is restricted to the generic data whose loss improvement between the pre-trained model (i) and the fine tuned model (ii) is the greatest. Finally, (iv) the training of the pre-trained model (i) is resumed on the selected data from stage (iii). Although CDS is a generic method applicable to any training objective, it enjoys additional properties when applied to generative models trained to maximize the (conditional) training likelihood. It can both be considered an importance sampling method and an influence function based selection method (Grangier & Iter, 2022).

Huang et al. (2006) cast weights estimation as a quadratic problem with a kernel function. Related to domain adaptation, the removal of label noise in the generic distribution has received attention with methods based on influence functions (Koh & Liang, 2017; Pruthi et al., 2020; Schioppa et al., 2022), data models (Ilyas et al., 2022; Jain et al., 2022), and data Shapley values (Ghorbani & Zou, 2019; Karlaš et al., 2022).

As an alternative to static weighting, the literature also explored dynamic weighting where the distribution over generic examples is adapted during training. The two primary strategies are reinforcement learning and direct optimization. Reinforcement learning does not assume that the specific task loss can be differentiated with respect to the weighting parameters. Instead, a parameterized model of the generic distribution is adjusted through reinforcement learning: the current model proposes generic distributions, and their reward is measured as the specific loss after a few steps of generic training over a proposal distribution (Kumar et al., 2019; Yoon et al., 2020; Zhu et al., 2020). On the other hand, direct optimization assumes a differentiable functional dependency between the weighting parameters and the specific training loss. This dependency can be derived through meta learning by unfolding the generic update (Ren et al., 2018; Hu et al., 2019; Shu et al., 2019; Zhang & Pfister, 2021): one gradient update step minimizing the weighted generic loss depends on the weighting parameters. The impact of this update can be evaluated by computing the post-update specific loss which can then be differentiated with respect to the weighting parameters. As an alternative to update unfolding, a bilevel formulation of the reweighting problem also allows for direct optimization (Franceschi et al., 2018). Our work builds upon this bilevel formulation.

Other research areas intersect with sample reweighting. Ho et al. (2019); Lim et al. (2019); Zoph et al. (2020) considered learning a distribution over training data augmentations. Curriculum learning visits successive training distributions based on training instance difficulty (Bengio et al., 2009; Kumar et al., 2010; Jiang et al., 2018; Saxena et al., 2019). Multi-task learning research has considered gradient projection to minimize negative interactions between tasks (Yu et al., 2020; Dery et al., 2020; Liu et al., 2021). Importance sampling for accelerated stochastic training (Zhao & Zhang, 2015; Katharopoulos & Fleuret, 2018) is also relevant.

3 Problem Setting

Classical machine learning assumes that the model is trained on data drawn from the distribution from which the test data will also be sampled from (Vapnik, 1999). Our setting is different and belongs to the field of transfer learning (Caruana, 1993; Thrun & Pratt, 1998). We are given two training sets, a large generic training set $\mathcal{D}_{\text{generic}}$ and small specific training set $\mathcal{D}_{\text{specific}}$. Only the latter set is representative of the test conditions. The large generic set can be leveraged as it might contain information related to the targeted specific. Its large scale allows more reliable statistical estimation and allows training higher capacity models.

Domain adaptation (Farahani et al., 2021), multi-task learning (Caruana, 1993), fine-tuning (Denevi et al., 2018) are transfer learning setups related to our setting, see Appendix A for a discussion on the relevant terminology. Importantly, we assume that the targeted task is known at the beginning of learning (which is not a necessity for fine tuning) and that only the specific task accuracy matters (unlike multi-task learning, which might also target high generic accuracy). While our main focus is targeting a single specific task, we discuss the case where the specific loss is a mixture over multiple targeted specific tasks in Appendix B.

Optimization-based Filtering. Large language and vision models have been shown to perform well on a wide range of test distributions. This stems from two combined causes: *(i)* a diverse training set which is a mixture over many domains and *(ii)* large capacity to fit each domain in the mixture. Therefore, there is an inherent tradeoff between capacity and generality. When one knows the application domain in advance, fitting a large web corpus without data selection will waste capacity on train examples far from the targeted domain. Ad-hoc selection techniques are commonplace for that purpose (filtering of captions by confidence in LAION, books are all you need for QA (Gunasekar et al., 2023), etc...). Our goal is to explore optimization-based techniques to complement widely used application-specific filtering techniques.

4 Methods

We aim to identify the parameters θ of a model that achieves good generalization performance (held-out likelihood) over the specific distribution. We are given a large generic training set $\mathcal{D}_{\text{generic}}$ and small specific training set $\mathcal{D}_{\text{specific}}$. We cast the generic training problem as the minimization of the weighted loss

$$\mathcal{L}_{\text{generic}}(\theta, \alpha) = \sum_{x \in \mathcal{D}_{\text{generic}}} w(x;\alpha)\ell(x;\theta)$$

where $w(x;\alpha)$ denotes a smaller, secondary *weighting neural network* which defines a distribution over $\mathcal{D}_{\text{generic}}$, i.e. $\forall x, w(x;\alpha) > 0$ and $\sum_{x\in\mathcal{D}_{\text{generic}}} w(x;\alpha) = 1$. The weighting network is parameterized by the parameters α. We propose to use a weighting network that is much smaller than the main model, so the dimension of α is typically smaller than that of θ. We denote the solution to generic training problem as

$$\theta^*(\alpha) \in \arg\min_{\theta} \mathcal{L}_{\text{generic}}(\theta, \alpha) \tag{1}$$

which depends on the weighting networks' parameters α. Our goal is to find the parameter of the weighting network such that the loss on the *specific* training set is minimal, i.e. minimizing,

$$\mathcal{L}_{\text{specific}}(\theta^*(\alpha)) := \sum_{x' \in \mathcal{D}_{\text{specific}}} \ell(x';\theta^*(\alpha)). \tag{2}$$

with respect to α, where the samples x' come from the specific training set.

4.1 Data Selection as a Bilevel Optimization Problem

The previous equations make it clear that finding the optimal weighting network is a bilevel optimization problem: with a fixed weighting network, the optimal parameters for the main model are found by minimizing the weighted loss over the generic dataset, $\mathcal{L}_{\text{generic}}$ (Equation 1). The optimal main model parameters θ^* depends explicitly on the weighting network parameters α: changing α changes the optimization problem in Equation 1 and its solution. The selection of α is driven by the specific set loss, $\mathcal{L}_{\text{specific}}$(Equation 2).

Equation 1 and Equation 2 form a *bilevel optimization problem* (Franceschi et al., 2018): the outer problem (Equation 2) depends implicitly on α through the solution to the inner problem (1). One of the strengths of such a bilevel formulation is that the weighting network must adapt to the main model: the question is to learn a weighting network such that the main model trained with that network leads to good specific performance. This has the potential to go beyond a simple model-agnostic scheme that would, for instance, build $w(x)$ based on the similarity between x and the specific set. While a large body of the literature is devoted to solving bilevel problems where the inner problem Equation 1 is convex in θ (Ghadimi & Wang, 2018; Arbel & Mairal, 2021), in our case, Equation 1 corresponds to the training problem of a neural network which is non-convex. This leads to several difficulties:

i) The $\arg\min$ in Equation 1 is not a single element since there are multiple minimizers. Therefore, the function $\theta^*(\alpha)$ is not properly defined.

ii) In order to use gradient-based methods to find the optimal α, we have to compute the approximate Jacobian of $\theta^*(\alpha)$. This is usually done using the implicit function theorem, which only applies when the loss function in equation 1 is locally convex and such property is hard to check in practice.

Furthermore, we want a method with a computational cost similar to the standard training of the main model. In other words, we have enough budget to solve Equation 1 only once: learning α and θ must be carried out synchronously. This has an important consequence: the bilevel methods that we study update α based on the current main model state θ and not on the optimal solution $\theta^*(\alpha)$. Hence, this is a slight deviation from the bilevel formalism. This also means that the weighting network adapts to the current state of the main model and, ideally, tries to up-weight generic data that is useful *at the current state of learning.* We explore online algorithms to solve the bilevel problem when the main model is large. These algorithms alternate θ and α updates and leverage the asymmetry in computation cost between evaluating the large main model and the small auxiliary weighting network.

Algorithm 1 Scalable, Online Bilevel Data Selection

```
Require: $\mathcal{D}_{\text{generic}}, \mathcal{D}_{\text{specific}}, b_{\text{small}}, b_{\text{large}}$    ▷ Training datasets, batch sizes.
  $\theta_0 \leftarrow$ main_model_initializer()
  $\alpha_0 \leftarrow$ weight_model_initializer()
  for $t = 1, \dots, T$ do
      ▷ Sample generic and specific batch.
      $B_{\text{generic}} \leftarrow$ sample($\mathcal{D}_{\text{generic}}, b_{\text{large}}$)
      $B_{\text{specific}} \leftarrow$ sample($\mathcal{D}_{\text{specific}}, b_{\text{small}}$)

      ▷ Sample generic sub-batches.
      $B_{\text{filtered}} \leftarrow$ filter($B_{\text{generic}}, \alpha_{t-1}, b_{\text{small}}$)
      $B'_{\text{generic}} \leftarrow$ sample($B_{\text{generic}}, b_{\text{small}}$)

      ▷ Inner and outer updates.
      $\theta_t \leftarrow$ update_main_model($B_{\text{filtered}}, \theta_{t-1}$)
      $\alpha_t \leftarrow$ update_weight_model($B'_{\text{generic}}, B_{\text{specific}}, \theta_t, \alpha_{t-1}$)
  end for
  return $\theta_T$    ▷ Trained main model.
```

4.2 Updating the main model

To update the main model, we fix α and do a step to minimize Equation 1. A first, natural idea would be to take a mini-batch of generic data B_{generic} of size b, compute the corresponding gradient $g = \frac{1}{b}\sum_{x \in B_{\text{generic}}} w(x;\alpha)\nabla_\theta \ell(x;\theta)$ and then use it to update θ, either implementing SGD by doing $\theta \leftarrow \theta - \eta \times g$ with $\eta > 0$ a learning rate, or by using it into a more involved optimizer like Adam. However, the computation of g with the previous equation can be wasteful when a significant fraction of the examples of B_{generic} are assigned small weights $w(x;\alpha)$. These examples do not contribute much to g while still requiring the expensive computation of their gradient $\nabla_\theta \ell(x;\theta)$.

To accelerate the optimization of θ, we leverage the asymmetry between the cost of evaluating the weighting network and the main model: computing $w(x;\alpha)$ only requires inference of a small network while computing $\nabla \ell(x;\theta)$ requires inference *and* back-propagation through a large network. We start by sampling a large batch $B^{\text{big}}_{\text{generic}}$ from the generic dataset and compute $w(x;\alpha)$ for each x in $B^{\text{big}}_{\text{generic}}$. From there we can take a smaller batch $B^{\text{small}}_{\text{generic}}$ from $B^{\text{big}}_{\text{generic}}$, either by sampling from the distribution defined by $w(x;\alpha)$ or by taking the examples with the highest $w(x;\alpha)$. The first option is an unbiased solution corresponding to importance sampling, while the second option is biased but observed to work better in practice. In both cases, we compute the gradient to update θ with uniform weights, using $g = \frac{1}{b} \sum_{x \in B_{\text{generic}}^{\text{small}}} \nabla_\theta \ell(x; \theta)$.

This proposed training method is ***sparse***: the samples in $B_{\text{generic}}^{\text{big}}$ have a sparse weight, only a fraction of them is non-zero. In the experiments, we prefix all methods that use this strategy with the sparse adjective.

4.3 Updating the weighting model

With scalability in mind, we only consider *stochastic* methods, i.e., that update the weighting network parameters α using only a mini-batch of specific data B_{specific} and a mini-batch of generic data B_{generic}. We consider three alternatives to update the weighting model.

Before describing alternative methods to update α, we summarize our approach in Algorithm 1. We denote sample(D, n) the set resulting from sampling n times uniformly from a set D. We denote filter(D, α, n) the result from either (a) sampling n times from D i.i.d relying on the distribution induced by the weighting model at α, or (b) selecting the top-n highest weighted examples from D. The batch sizes $b_{\text{small}}, b_{\text{large}}$ are hyper-parameters selected through validation.

4.3.1 One gradient step unrolling - differentiable data selection (DDS)

This method updates the weighting network by doing a descent step on the loss

$$\mathcal{L}(\alpha) = \sum_{x' \in B_{\text{specific}}} \ell'(x'; u(\theta, \alpha)) \text{ with } u(\theta; \alpha) = \theta - \rho \times \sum_{x \in B_{\text{generic}}} w(x; \alpha) \nabla_\theta \ell(x; \theta), \tag{3}$$

which corresponds to the value of the specific loss on the mini-batch B_{specific} after a gradient descent step for θ on the generic mini-batch B_{generic} using the current weights. It is similar to (Wang et al., 2020). The idea behind this method is that $u(\theta, \alpha)$ is a reasonable approximation to $\theta^*(\alpha)$. This method requires back-propagating through a gradient descent step, which requires only a little overhead compared to a standard gradient computation. In the limit where the step size ρ in the gradient update $u(\theta, \alpha)$ goes to 0, we see that $\mathcal{L}(\alpha) \simeq \rho \langle g_{\text{specific}}, g_{\text{generic}} \rangle$, with $g_{\text{specific}} = \sum_{x' \in B_{\text{specific}}} \nabla \ell'(x'; \theta)$ and $g_{\text{generic}} = \sum_{x \in B_{\text{generic}}} w(x; \alpha) \nabla \ell(x, \theta)$. Hence, the loss $\mathcal{L}$ approximately measures the alignment between specific and generic gradients. Taking derivatives gives $\nabla \mathcal{L}(\alpha) \simeq \rho \sum_{x \in B_{\text{generic}}} \langle g_{\text{specific}}, \nabla \ell(x, \theta) \rangle \nabla w(x; \alpha)$.

4.3.2 Stochastic Bilevel Algorithm (SOBA)

We also implement the SOBA method of (Dagréou et al., 2022), which is a scalable method to solve the bilevel problem, developed in a setting where the inner function (Equation 1) is convex. This algorithm approximates a gradient descent on $h(\alpha) = \mathcal{L}_{\text{specific}}(\theta^*(\alpha))$. The chain rule gives $\nabla h(\alpha) = \frac{\partial \theta^*}{\partial \alpha} \nabla \mathcal{L}_{\text{specific}}(\theta^*(\alpha))$. The optimum $\theta^*(\alpha)$ satisfies the first order condition $\nabla_\theta \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha) = 0$. Under the assumption that the Hessian $\nabla^2_{\theta\theta} \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha)$ is invertible, the implicit function theorem applied to the previous equation gives $\frac{\partial \theta^*}{\partial \alpha} = -\nabla^2_{\alpha\theta} \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha) \left[\nabla^2_{\theta\theta} \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha)\right]^{-1}$, which overall yields $\nabla h(\alpha) = -\nabla^2_{\alpha\theta} \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha) \left[\nabla^2_{\theta\theta} \mathcal{L}_{\text{generic}}(\theta^*(\alpha), \alpha)\right]^{-1} \nabla \mathcal{L}_{\text{specific}}(\theta^*(\alpha))$. SOBA approximates this quantity in two ways: first, $\theta^*(\alpha)$ is replaced by the current iterate θ in the above gradient. Second, in addition to θ and α, SOBA has an additional variable v of the same size as θ that keeps track of the quantity $-\left[\nabla^2_{\theta\theta} \mathcal{L}_{\text{generic}}(\theta, \alpha)\right]^{-1} \nabla_\theta \mathcal{L}_{\text{specific}}(\theta)$. This is done using the stochastic iterations $v \leftarrow v - \eta \times dv$ with $dv = \sum_{x \in B_{\text{generic}}} w(x; \alpha) \nabla^2 \ell(x; \theta) v + \sum_{x' \in B_{\text{specific}}} \nabla \ell'(x'; \theta)$. The first part in dv is a Hessian-vector product that can be computed efficiently at a cost similar to that of a gradient (Pearlmutter, 1994). Then, the parameters α are moved in the direction $d\alpha = \sum_{x \in B_{\text{generic}}} \langle \nabla \ell(x; \theta), v \rangle \nabla w(x; \alpha)$, which is a stochastic approximation of $\nabla^2_{\alpha\theta} \mathcal{L}_{\text{generic}}(\theta, \alpha) v$, which is itself an approximation of $\nabla h(\alpha)$.

4.3.3 Aligned NOrmalized GRADient (Anograd)

We derive Anograd (Aligned NOrmalized GRADient) as a variant of DDS which relies on steepest descent (Boyd & Vandenberghe, 2004). We apply the steepest descent algorithm to the specific loss

$\theta \to \mathcal{L}_{\text{specific}}(\theta)$. We recall that the steepest normalised descent direction according to the Euclidean norm $\|\cdot\|$ for the specific loss is

$$\Delta\theta_{\text{nsd}} = \arg\min_{v}\{v^\top \nabla_\theta \mathcal{L}_{\text{specific}}(\theta) : \|v\| = 1\} \tag{4}$$

This direction aligns with the opposite gradient when v is not further constrained. In our case, θ updates should correspond to gradient descent updates on the weighted generic loss. We therefore constraints θ updates to decompose as an afine combination of individual generic example gradients, i.e. $\sum_{i=1}^{n_{\text{g}}} a_i \nabla \ell(x_i^{\text{g}}, \theta)$ where $\mathcal{D}_{\text{generic}}$ is denoted as $\{x_i^{\text{g}}\}_{i=1}^{n_{\text{g}}}$ and $a_i > 0, \forall i$. Therefore, we need to solve Equation 4 with the constraint $v \in \mathcal{V}$, with $\mathcal{V} = \left\{\sum_{i=1}^{n_{\text{g}}} a_i \nabla \ell(x_i^{\text{g}}, \theta), \forall a_i \geq 0\right\}$. This amounts to solving

$$\min_a \left(\frac{\sum_{i=1}^{n_{\text{g}}} a_i \nabla \ell(x_i^{\text{g}}, \theta)}{\|\sum_{i=1}^{n_{\text{g}}} a_i \nabla \ell(x_i^{\text{g}}, \theta))\|}\right)^\top \nabla_\theta \mathcal{L}_{\text{specific}}(\theta)$$

which itself is equivalent to solve $\min_a$ $\text{cosine}\left(\sum_{i=1}^{n_{\text{g}}} a_i \nabla \ell(x_i^{\text{g}}, \theta),\ \nabla_\theta \mathcal{L}_{\text{specific}}(\theta)\right)$ We now parameterize a as the output of the weighting network and introduce the loss, $\mathcal{L}_{\text{anograd}}(\theta, \alpha) = \text{cosine}\left(\nabla_\theta \mathcal{L}_{\text{generic}}(\theta, \alpha),\ \nabla_\theta \mathcal{L}_{\text{specific}}(\theta)\right)$. The anograd method performs gradient descent on that loss to update α. Like for DDS and Soba, we perform a single step before updating θ. For scalability we rely on stochastic (batch) estimates for both both terms in the cosine. Compared to DDS, the normalization in anograd reduces the benefit of up-weighting generic examples with high gradient norm.

5 Experiments & Results

Our experiments focus on three application domains: language modeling, machine translation and image classification. Before introducing our experimental setup and discussing our results on each domain, we describe the baselines we considered.

5.1 Evaluated Alternative Methods

For our empirical comparison, we first consider two common, simple methods which do not rely on data selection. We call *baseline* pretraining on the generic training set followed by fine tuning on the specific set. We call *mixing* pretraining on a mix of generic and specific data. Each training batch contains a fixed fraction of specific data. This fraction is selected by validation.

Our first baseline is *contrastive data selection*, CDS, as described in section 2. As we do for all data selection method, we consider further fine tuning the final CDS model on the specific training set.

We also consider a *domain classifier*. In that case, a simple model is pretrained on a binary classification problem to distinguish between generic and specific training examples. The model has the same architecture as the weighting model we use with bilevel methods and it minimizes the binary cross entropy on batches with the same proportion of specific and generic data. This model can estimate the probability that an example belongs to the specific set and is applied to restrict the generic set to the data with the highest estimates. We can train a model on this restricted set and later fine tuning on the specific data.

Closer to our bilevel selection methods, we evaluate learning to re-weight, LTR (Ren et al., 2018) and meta-weight net (Shu et al., 2019)). Learning to re-weight is similar to the DDS approach we presented in Section 4 except it does not maintain a weighting model. Instead, at each step, the model considers a uniform distribution over the generic batch. It then computes the gradient of this flat weighting as free parameters with the outer update, Equation 2. This single step updates from uniform is then used to reweight the generic loss and update the main model, Equation 1. Compared to our work, this method does not persist a weighting model across steps and does not allow learning complex distributions. The lack of weighting model is also less efficient since a generic example x cannot be discarded without evaluating the main model and its gradient at x.

Meta-weight net is a particular, simple case of DDS in which the weight model takes as input a single scalar for each example: the example loss, i.e. $w(x; \alpha) = \text{mlp}(\ell(x; \theta); \alpha)$. This parameterization is sensible for some applications, e.g. loss based up-weighting is a common approach for active learning in classification problems

Table 1: Model architectures

Language Model

Main model: Transformer decoder with 12 layers, 8 attention heads, residual dimension of 256, feed-forward latent dimension of 1,024.
Weight model: Convolutional network with 2 layers followed by mean pooling, latent dimension of 128.

Translation Model

Main model: Transformer with 6 encoder layers and 6 decoder layers, 16 attention heads, residual dimension of 1,024, feed-forward latent dimension of 4,096.
Weight model: Embedding layer of dimension 32 followed by an MLP with a latent dimension of 128.

Image Classifier

Main model: Dual encoder clip model with ResNet 50 for images (224x224) and an multi-layer perceptron (2 latent layers with dim. 768) on top of sentence BERT for text.
Weight model: Convolutional network over 32x32 images with 4 layers of dimension 32, 32, 32 and 64.

with little intrinsic uncertainty (Settles, 2009). Loss values can be indicative of an example difficulty and loss based up-weighting might accelerate learning. However, an example loss seems to be orthogonal to the example value for domain transfer.

5.2 Language Modeling

Our language modeling (LM) experiments relies on two datasets, the C4 dataset (Raffel et al., 2019) is used as the generic set and the RCV1 (Lewis et al., 2004) dataset is used as the specific set. C4 is a dataset of English language web pages from common crawl (Patel, 2020), while RCV1 consists of newswire stories from Reuters. This setup is representative of a generic large corpus spanning different types of examples (c4) while the specific task contains an homogeneous set of examples from the same domain and from the same source (RCV1). In our setup, we use 30m examples from C4 and 10k examples from RCV1.

Method	Pre-train	Fine-tune
Baseline	1.198 ± 0.003	0.864 ± 0.002
Mixing	0.861 ± 0.002	0.847 ± 0.002
CDS	1.067 ± 0.005	0.867 ± 0.002
Domain classif.	1.098 ± 0.002	0.894 ± 0.003
MetaWeightNet	1.212 ± 0.004	0.868 ± 0.003
LTR	1.156 ± 0.002	0.879 ± 0.002
Sparse DDS	1.039 ± 0.004	② 0.824 ± 0.003
Sparse Anograd	1.034 ± 0.002	② 0.824 ± 0.002
Sparse SOBA	1.019 ± 0.003	① 0.820 ± 0.002

Table 2: Language modeling: Log-perplexity (negative log-likelihood per byte) on specific (Reuters). Circled numbers indicate the best results. Numbers after the ± are the standard deviation between 5 runs. The adjective *sparse* refers to methods that use the sparse batch trick described in Sec. 4.2.

Our language model is a byte-level language model based on the transformer decoder architecture (Vaswani et al., 2017). Although subword language models are more common than byte-level ones (Sennrich et al., 2016; Al-Rfou et al., 2019), we rely on bytes to avoid our out-of-domain generalization results to be contaminated by the mismatch between the evaluation data and the tokenizer training data (Rust et al., 2021). The weighting network is a small convolutional network. Table 1 gives architectural details. We also use the same architecture for the domain classifier baseline. We report performance in terms of log-perplexity, i.e. negative log likelihood. Our implementation is derived from the language model of the Flax library (Heek et al., 2020).

Table 2 reports the results of our language modeling experiments. In general, domain adaptation is beneficial in our setting. The only method trained exclusively on c4 (baseline without fine-tuning) is much worse than all alternatives except for MetaWeightNet. Before pretraining, mixing is the only method which directly applies updates from the specific training data and it performs best. The other methods only emphasize part of the generic set without applying specific updates during pretraining. This

Table 3: Language modeling: Log-perplexity on specific for Pile domains.

Method	arxiv	europarl	freelaw	gutenb.	opensub.	openweb.	pmed abs	stackex.	wikipedia
Base	1.438	2.219	1.555	1.365	1.277	1.220	1.088	1.313	1.103
+ ft.	0.898	0.993	0.603	0.488	1.058	1.180	0.870	1.039	0.877
Mixing	0.909	1.019	0.606	0.487	1.067	1.153	0.874	1.049	0.874
+ ft.	0.899	1.081	0.600	0.478	1.059	1.156	0.860	1.042	0.865
CDS	1.216	1.981	1.284	1.253	1.193	1.154	0.829	1.100	0.944
+ ft.	① 0.861	0.977	0.614	0.482	② 1.039	③ 1.131	0.791	① 0.971	③ 0.823
Classifier	1.293	1.608	1.133	1.202	1.266	1.139	0.787	1.159	0.914
+ ft.	0.920	② 0.892	③ 0.582	0.481	1.066	① 1.125	① 0.765	0.998	② 0.807
S. DDS	1.231	1.868	1.184	1.285	1.288	1.290	0.828	1.112	0.988
+ ft.	② 0.867	0.948	0.580	① 0.477	1.104	1.262	0.792	② 0.977	0.839
S.Anograd	1.219	1.659	1.237	1.274	1.193	1.150	0.814	1.132	0.989
+ ft.	③ 0.871	③ 0.895	② 0.580	① 0.477	③ 1.042	1.133	③ 0.784	0.988	0.838
S. SOBA	1.210	1.582	1.124	1.296	1.184	1.149	0.803	1.134	0.908
+ ft.	0.872	① 0.883	① 0.579	② 0.480	① 1.035	② 1.128	② 0.779	③ 0.989	① 0.803

emphasis already show a benefit at pretraining time. More importantly, this benefit is complementary to fine tuning (Iter & Grangier, 2021) and these methods yield better results that mixing+fine-tuning. Among them, bilevel methods perform best, with SOBA giving the highest held-out specific likelihood.

We perform additional language modeling experiments with different domains. We take 9 domains from (Gao et al., 2021) and rely on 10k specific document for each domain. The generic set (c4 dataset) and the model architectures are the same as in the previous experiments. Results are given in Table 3. Data selection methods show that it is helpful to emphasize part of the generic set and that this emphasis is complementary to the benefit of fine tuning. The benefit varies across domains. For instance openweb is similar to the generic set c4 and only modest gains are observed, while freelaw contains legal proceedings whose domain is surely relatively rare in c4. Among methods, CDS and classifier provides a strong benefit for some datasets, but only SOBA consistently ranks among the best methods.

5.3 Machine Translation

Method	Pre-train		Fine-tune	
	BLEU	loss	BLEU	loss
Baseline	27.63	2.56	34.06	2.53
Mixing	31.34	2.60	33.11	2.69
CDS	34.14	2.53	34.25	2.53
Domain classif.	35.56	2.37	① 38.03	2.35
MetaWeightNet	26.81	2.59	33.34	2.53
LTR	28.60	2.73	31.15	2.71
Sparse DDS	33.53	2.46	35.83	2.44
Sparse Anograd	36.06	2.41	② 37.28	2.40
Sparse SOBA	34.23	2.39	③ 37.16	2.38

Table 4: Machine translation: BLEU and loss on specific (newstest2020).

Our machine translation (MT) experiments learn a translation model from English into German. They rely on two datasets: our generic set is the Paracrawl dataset (Release 1 for WMT 2018) with 36m sentence pairs (Bañón et al., 2020). Our specific set concatenates the WMT newstest sets (2009–2019) with source original English sentences, which amounts to 10,015 sentence pairs (Akhbardeh et al., 2021). We use the 2020 newstest data (1,997 sentences) as our validation set and leave the 2021 newstest data (1,418 sentences) as our test set. Our generic set is therefore a large crawled set with different types of text and varying translation quality while the specific set is a small set from a single domain with high quality translation.

Our translation system is a sub-word model based on the transformer encoder-decoder architecture. For the weighting network and the domain classifier we compose a shared embedding layer for source and target and apply a multi-layered perceptron on the contatenated averaged embeddings of the source and target sentences.

Table 1 gives architectural details. Our evaluation relies on BLEU scores (Papineni et al., 2002) for beam-4 decodes. We also reports some results in terms of loss (i.e. negative log likelihood with label smoothing strength of 0.1). Our implementation is derived from the translation model of the Flax library (Heek et al., 2020).

Table 4 reports the results of our machine translation experiments. In that case, SOBA and Anograd provide a strong improvement over the baseline, both before (more than +7 BLEU) and after fine tuning (more than +3 BLEU). However in this setting, the domain classifier is even more effective. Both for language modeling and for machine translation, we remark that MetaWeightNet performs poorly. MetaWeightNet predicts the selection weight from the loss on the example. This is a common strategy in active learning for classification (Settles, 2009). This notably assumes that the loss per example is indicative of how well the model perform on them. However, in the case of language modeling and translation, the loss per example also reflects the intrinsic entropy of the examples which varies greatly across examples. It therefore seems difficult to use the loss the sole feature for data selection for these tasks. Comparing the results of LTR and DDS is also interesting as the methods are similar. DDS maintains a weighting model across steps, while LTR just adapt the distribution for each batch and does not persist cross-steps selection parameters. The benefit of DDS tells that the stability across steps and the lesser dependency on the batch size are important.

5.4 Image Classification

Method	Pre-train		Fine-tune	
	Acc.	Loss	Acc.	Loss
Baseline	41.1	2.694	54.9	1.902
Mixing	55.1	1.928	③ 55.1	1.928
CDS	42.3	2.512	② 55.2	1.957
Domain classif.	44.1	2.571	① 57.5	1.949
MetaWeightNet	35.5	2.743	43.9	2.351
LTR	36.4	2.712	44.9	2.364
Sparse DDS	40.5	2.609	53.2	2.067
Sparse Anograd	41.4	2.563	53.6	2.055
Sparse SOBA	41.1	2.622	53.9	2.057

Table 5: Image Classification: Accuracy on specific (ImageNet67).

Our vision setup performs contrastive training over image and captions – CLIP (Radford et al., 2021) – for generic training and image classification for specific training. Specifically, contrastive learning should select the correct caption within a large set of random captions. This approach also allows to perform classification by representing classes as captions of the form "a photo of a <class name>" and letting the model infer the most appropriate caption within that set. As datasets, we rely on yfcc15m (Radford et al., 2021) for generic training (14.9m image/caption pairs) and ImageNet67 (Eshed, 2020) dataset for the specific task. Imagenet 67 consists in 67 high level classes over Imagenet (Deng et al., 2009), e.g. building, person, fish, dog... Like for other experiments, we consider a setup with limited specific data and take 2,010 specific examples, 30 per class, for training. Held-out evaluation is performed with 50 images per class.

For our CLIP model, the image branch is a Resnet 50 (He et al., 2016) while the text branch applies an MLP over precomputed sentence embeddings from Sentence BERT (Reimers & Gurevych, 2019). Training applies contrastive learning only over alternative captions: for the generic loss, we consider a buffer of past captions as negatives; for the specific loss, we consider all the other class captions as negatives. Our weighting network is a small convolutional network over low resolution images (32x32). Table 1 gives architectural details.

Table 5 reports the results of our image classification experiments. Unlike for our text experiments, the benefit of data selection is limited for this task. After fine-tuning, only the CDS and domain classifier methods outperform the baseline model. The bilevel data selection methods do not outperform the baseline method. We shed light on the cause of this poor performance in our further analysis in Section 6.3.

Table 6: Does the weight model's trajectory correspond to a curriculum? Pre-train LM log perplexity on Reuters for different trajectories.

Weight model trajectory	log-perplexity
SOBA curriculum	1.047
Final weighting	1.044
Shuffled weighting	1.055

Table 7: Alternative sampling strategies to filter the generic batch. Pre-train LM log perplexity on Reuters.

Sampling strategy	log-perplexity
Importance Sampling	2.038
Sampling without replacement	1.040
Selecting the highest weights	1.227

Table 8: Specific & Generic Acceleration Rates

Task		SAR	GAR
Language modeling	(c4, reuters)	86.2%	69.4%
Machine translation	(paracrawl, newstest)	78.1%	68.2%
Image classification	(yfcc15m, imagenet67)	50.3%	49.8%

6 Analysis

6.1 Learning a Distribution vs Learning a Curriculum

Algorithm 1 produces a sequence of main model's parameters θ_t and weighting model's parameters α_t, that go towards the solution of the bilevel problem (2). We investigate whether the weighting model's parameters correspond to a *curriculum*: does the evolution of the weighting parameters α_t adapt to the particular data needed at each step, helping the model perform better than a fixed weighting scheme?

We are in the LM task setup described in Section 5.2, except that we use a smaller "large batch size" $B^{\text{big}}_{\text{generic}}$ (see Section 4.2). We run Algorithm 1 with SOBA to obtain a sequence θ_t, α_t. We then compare this setting with two new training runs with standard ERM using different data weighting:
-*Final weighting:* a new main model is trained with fixed weighting from the weighting model α_T.
-*Shuffled weighting:* a new main model is trained with a random permutation σ of the weights $\alpha_{\sigma(t)}$.
Table 6 shows that SOBA's curriculum is not beneficial compared to the fixed final weighting scheme on this task. The lesser performance of shuffled weighting certainly highlight poor weighting from early α_t. Results reported in this section do not match Section 5.2 because of a smaller $B^{\text{big}}_{\text{generic}}$ was used in this ablation.

6.2 Big Batches: Importance Sampling vs Filtering

In Algorithm 1, we denote filter(B, α, n) the operation resulting in a smaller sub-batch of size n starting from the generic batch B using the weighting network parameterized by α. To get an unbiased estimate of the re-weighted generic loss, one can apply *importance sampling* and sample (with replacement) from the weight distribution induced by α on B. Alternatively one can instead sample *without replacement* from that distribution or restrict the batch B to its *highest weighted* elements. The last two alternative are biased. Nevertheless, our results in Section 5 uses sampling without replacement.

Table 7 justifies this choice. Basically, we observe that the learned weighted distribution is concentrated along few examples which yield importance sampling batches to contain less diverse sets than when sampling with replacement. Similarly, cutting the tail of the distribution (highest weights selection) drop lower weighted – but still helpful – examples. These experiments illustrate that gradient-based estimates fail to account for the long term benefit of a more diverse training set. Although sampling with replacement alleviates this issue, more principled solutions should be investigated in future work.

6.3 On the Discriminative Power of Gradient Aligments

Our experiments highlight that bilevel optimization for data selection performs differently across tasks. We explore if a simple diagnostic could help understand these differences. Our method considers a base model θ_t

trained on the generic distribution for t steps. We take a diagnostic batch B_{mix} which blends unseen generic and specific data in equal proportion. We want to verify how the weighting model on the mixed data would move away from a uniform weighting scheme in an outer update. We want to observe whether the weighting model would increase the weights of specific examples if some of these were "hidden" within the generic set.

For DDS and Anograd, increase or decrease in weights depends on the alignment between individual example gradients from $x \in B_{\text{mix}}$ and the expected gradient on the training specific batch B_{specific}.

$$a(x, B_{\text{specific}}) = \nabla_\theta \ell(x,\theta)^\top \; \nabla_\theta \ell(B_{\text{specific}}, \theta)$$

between individual example gradients from $x \in B_{\text{mix}}$ and the expected gradient on the training specific batch B_{specific}, denoted as $\ell(B_{\text{specific}}, \theta) := \frac{1}{|B_{\text{specific}}|}\sum_{x\in B_{\text{specific}}} \ell(x,\theta)$. We then normalize the batch gradient and define,

$$a_{\text{norm}}(x, B_{\text{specific}}) = \nabla_\theta \ell(x,\theta)^\top \; \frac{\nabla_\theta \ell(B_{\text{specific}}, \theta)}{\|\nabla_\theta \ell(B_{\text{specific}}, \theta)\|}.$$

This normalization allows to take an example x and verify whether its gradient aligns better with the specific batch gradient than with the generic batch gradient, i.e.

$$a_{\text{norm}}(x, B_{\text{specific}}) > a_{\text{norm}}(x, B_{\text{generic}}).$$

We report the rate at which this inequality is true for specific examples,

$$\text{SAR} = \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D}_{\text{specific}}\\ B_{\text{specific}}\sim\text{Batch}(\mathcal{D}_{\text{specific}})\\ B_{\text{generic}}\sim\text{Batch}(\mathcal{D}_{\text{generic}})}} \mathbb{1}\left\{a_{\text{norm}}(x, B_{\text{specific}}) > a_{\text{norm}}(x, B_{\text{generic}})\right\}$$

We call this measure the Specific Acceleration Rate, SAR. We would like this rate to be high, meaning that, according to the Taylor approximation of the loss, updates collected from a batch of specific examples should improve the loss on a given specific examples faster than updates collected from a generic batch. Symmetrically, we define the Generic Acceleration Rate,

$$\text{GAR} = \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{D}_{\text{generic}}\\ B_{\text{specific}}\sim\text{Batch}(\mathcal{D}_{\text{specific}})\\ B_{\text{generic}}\sim\text{Batch}(\mathcal{D}_{\text{generic}})}} \mathbb{1}\left\{a_{\text{norm}}(x, B_{\text{generic}}) > a_{\text{norm}}(x, B_{\text{specific}})\right\}$$

It is also desirable that this rate is high. When SAR, GAR are close to chance (50%), there are two possible explanations, (i) either generic and specific batches have the same effect on the model, meaning that data selection is unlikely to be helpful since training on generic is already as good as training on specific for the purpose of minimizing the specific loss; (ii) alternatively, the linear approximation (order 1 Taylor expansion) does not help discriminating between the effect of generic and specific examples on the specific loss. In that later case, such a learning problem will be a challenge for bilevel optimization methods where gradient alignments indicates which part of the dataset to upweight.

Table 8 reports SAR and GAR for our results. These results are indicative of the empirical benefit of bilevel optimization methods for data selection. Language modeling where DDS, Anograd and SOBA are advantageous, has the highest SAR. Conversely, our image classification problem shows near random SAR, GAR in line with the poor performance on bilevel methods on this problem. We therefore consider that measuring SAR/GAR can be a simple but informative diagnostic to assess the potential benefit of bilevel methods on a new problem.

6.4 Re-using Weighting Strategies with Larger Scale Models

The weighting network is trained by solving the bilevel problem (2), where the loss function ℓ depends on the model's architecture. We investigate whether a weighting network learned with a small model can be re-used out-of-the-box to train a large model and get good performances on the specific set. The weighting network is frozen: the large model is trained by solving $\min_\theta \sum_{x\in\mathcal{D}_{\text{generic}}} w(x;\alpha)\ell(x;\theta)$ with α fixed to the final parameters of the weighting network trained with the small model. We perform this experiment on

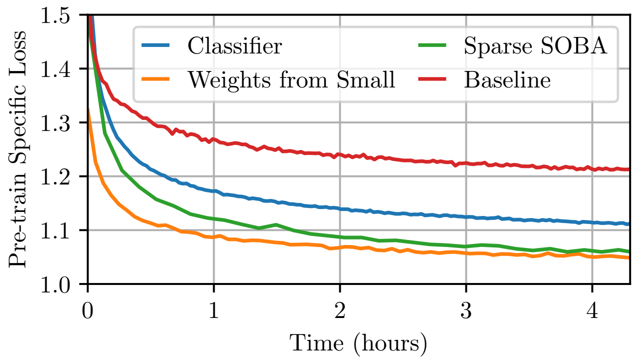

Figure 1: Specific loss as a function of time for the scaling experiment. Sparse-SOBA — which learns weights on the fly — accelerates learning. Using frozen weights learned with a small model leads to even faster training because this removes the cost of learning the weights. The training acceleration is significant. The loss value of 1.3 is reached 8.5 (resp. 3.9) times faster by the frozen weights (resp. Sparse SOBA) method than the baseline. The loss value of 1.25 is reached 21 (resp. 5) times faster by the frozen weights (resp. Sparse SOBA) method than the baseline. Note that the Sparse-SOBA method eventually gets better than the frozen weights method, as reported in Table 10, but this happens after a longer training time.

Table 9: Small and base model architectures for scaling up the language modeling task.

Model	Layers	Res. dim.	ff dim.	# Params
Small	4	128	512	824.064
Large	12	256	1024	9.530.880

Table 10: Results of the scaling experiment

Model	Method	Pre-train
Small	Sparse SOBA	1.292
Large	Baseline	1.197
Large	Sparse SOBA	1.018
Large	Weights from Small	1.034

the language modeling task (Section 5.2), where the small model and large model architectures are specified in Table 9; the large model's architecture is the same as in Section 5.2, and it has about ten times more parameters. We observe that the weighting network learned with the small model transfers to the large architecture and leads to a large decrease in the loss on the specific set, which is only slightly worse than using the Sparse SOBA method on the large model itself. The weighting network learned at a small scale can seamlessly be used at a larger scale and lead to significant performance improvement on the specific set.

In order to have a different perspective on the improvements provided by the reweighting methods, we display the training curves for that experiment in Figure 1. We see that the proposed methods lead to significantly faster training on the specific set.

7 Conclusions

This work studies bilevel optimization for learning training distributions. We consider the setup where a model is trained from two training sets, a large generic set and a small specific set, where only the later is representative of the test conditions. We propose a scalable algorithm that learns a training distribution over the generic data such that the loss on the specific set is minimized. We showed that our formulation gathers independently-proposed gradient-based methods for data selection under a common framework. We introduced an algorithm that enables streaming through the generic dataset quickly by examining most of the generic samples with only an inexpensive small auxiliary model. This work reported a comprehensive and realistic empirical comparison of data selection strategies across language modeling, machine translation and computer vision. We studied the conditions in which gradient-based data selection is effective and propose a diagnostic based on gradient alignment to efficiently assess these conditions.

Our work also delineates interesting questions for future work. Conceptually, we observe that gradient-based selection methods fail to reward properly the diversity of the selected samples (Section 6.2), which deserves further theoretical study. Empirically, the complementarity between fine-tuning and generic data selection highlights that the updates collected from re-weighted generic training and from specific training are different. The existence of complementary updates and their exploitation might also be possible even when one is presented with a single monolithic training distribution.

The experiments in this paper have a bigger scale than most experiments found in the data reweighting litterature (Ren et al., 2018; Shu et al., 2019). However, our methods are not applied to what is nowadays considered "large" models. The scaling ablation (subsection 6.4) suggests that the proposed methods would be well-suited for large-scale models, and we plan to conduct experiments on such models in the future.

References

Roee Aharoni and Yoav Goldberg. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7747–7763, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.692. URL `https://aclanthology.org/2020.acl-main.692`.

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pp. 1–88, Online, November 2021. Association for Computational Linguistics. URL `https://aclanthology.org/2021.wmt-1.1`.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3159–3166, 2019.

Michael Arbel and Julien Mairal. Amortized implicit differentiation for stochastic bilevel optimization. *arXiv preprint arXiv:2111.14580*, 2021.

Marta Bañón, Pinzhen Chen, Barry Haddow, Kenneth Heafield, Hieu Hoang, Miquel Esplà-Gomis, Mikel L Forcada, Amir Kamran, Faheem Kirefu, Philipp Koehn, et al. Paracrawl: Web-scale acquisition of parallel corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4555–4567, 2020.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman (eds.), *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pp. 41–48. ACM, 2009. doi: 10.1145/1553374.1553380. URL `https://doi.org/10.1145/1553374.1553380`.

Paul N Bennett, Susan T Dumais, and Eric Horvitz. Inductive transfer for text classification using generalized reliability indicators. In *Proceedings of the ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, 2003.

Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(9), 2009.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora

Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2021.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

R Caruana. Multitask learning: A knowledge-based source of inductive bias1. In *Proceedings of the Tenth International Conference on Machine Learning*, pp. 41–48. Citeseer, 1993.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Mathieu Dagréou, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. In *NeurIPS*, 2022.

Giulia Denevi, Carlo Ciliberto, Dimitris Stamos, and Massimiliano Pontil. Learning to learn around a common mean. *Advances in Neural Information Processing Systems*, 31, 2018.

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*, pp. 248–255. Ieee, 2009.

Lucio M Dery, Yann Dauphin, and David Grangier. Auxiliary task update decomposition: The good, the bad and the neutral, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Noam Eshed. Novelty detection and analysis in convolutional neural networks. Master's thesis, Cornell University, 2020.

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pp. 877–894, 2021.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pp. 1568–1577. PMLR, 2018.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1180–1189, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/ganin15.html`.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027, 2021. URL `https://arxiv.org/abs/2101.00027`.

Saurabh Garg, Yifan Wu, Sivaraman Balakrishnan, and Zachary Lipton. A unified view of label shift estimation. *Advances in Neural Information Processing Systems*, 33:3290–3300, 2020.

Saeed Ghadimi and Mengdi Wang. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pp. 2242–2251. PMLR, 2019.

David Grangier and Dan Iter. The trade-offs of domain adaptation for neural language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 3802–3813. Association for Computational Linguistics, 2022. URL `https://aclanthology.org/2022.acl-long.264`.

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.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*, 2020.

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 (CVPR)*, June 2016.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for jax, 2020. *URL http://github.com/google/flax*, 1, 2020.

Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pp. 2731–2741. PMLR, 2019.

Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. Learning data manipulation for augmentation and weighting. *Advances in Neural Information Processing Systems*, 32, 2019.

Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola. Correcting sample selection bias by unlabeled data. *Advances in neural information processing systems*, 19, 2006.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data. In *ICML*, 2022.

Dan Iter and David Grangier. On the complementarity of data selection and fine tuning for domain adaptation. *CoRR*, abs/2109.07591, 2021. URL `https://arxiv.org/abs/2109.07591`.

Saachi Jain, Hadi Salman, Alaa Khaddaj, Eric Wong, Sung Min Park, and Aleksander Madry. A data-based perspective on transfer learning. *arXiv preprint arXiv:2207.05739*, 2022.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pp. 2304–2313. PMLR, 2018.

Bojan Karlaš, David Dao, Matteo Interlandi, Bo Li, Sebastian Schelter, Wentao Wu, and Ce Zhang. Data debugging with shapley importance over end-to-end machine learning pipelines. *arXiv preprint arXiv:2204.11131*, 2022.

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

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5637–5664. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/koh21a.html`.

Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pp. 5468–5479. PMLR, 2020.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. Reinforcement learning based curriculum optimization for neural machine translation. *arXiv preprint arXiv:1903.00041*, 2019.

M. Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL `https://proceedings.neurips.cc/paper/2010/file/e57c6b956a6521b28495f2886ca0977a-Paper.pdf`.

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *AAAI*, volume 1, pp. 3, 2008.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5(Apr):361–397, 2004. URL `http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf`.

Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32, 2019.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.

Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part II*, volume 11206 of *Lecture Notes in Computer Science*, pp. 185–201. Springer, 2018.

Nada Matic, Isabelle Guyon, J Denker, and Vladimir Vapnik. Writer-adaptation for on-line handwritten character recognition. In *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 187–191. IEEE, 1993.

Robert C. Moore and William Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pp. 220–224, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `https://aclanthology.org/P10-2041`.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Jay M Patel. Introduction to common crawl datasets. In *Getting Structured Data from the Internet*, pp. 277–324. Springer, 2020.

Barak A Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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*, pp. 8748–8763. PMLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL `https://arxiv.org/abs/1908.10084`.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pp. 4334–4343. PMLR, 2018.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3118–3135, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.243. URL `https://aclanthology.org/2021.acl-long.243`.

Shreyas Saxena, Oncel Tuzel, and Dennis DeCoste. Data parameters: A new family of parameters for learning a differentiable curriculum. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/926ffc0ca56636b9e73c565cf994ea5a-Paper.pdf`.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8179–8186, 2022.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*, 2019.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL `https://aclanthology.org/P16-1162`.

Burr Settles. Active learning literature survey. 2009.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, 2019.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer, 1998.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1400–1410, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1147. URL `https://aclanthology.org/D17-1147`.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 133–143, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6314. URL `https://aclanthology.org/W18-6314`.

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Jaime Carbonell, and Graham Neubig. Optimizing data usage via differentiable rewards. In *International Conference on Machine Learning*, pp. 9983–9995. PMLR, 2020.

Jinsung Yoon, Sercan Arik, and Tomas Pfister. Data valuation using reinforcement learning. In *International Conference on Machine Learning*, pp. 10842–10851. PMLR, 2020.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting few-sample BERT fine-tuning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=cO1IH43yUF`.

Zizhao Zhang and Tomas Pfister. Learning fast sample re-weighting without reward data. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 705–714. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00076. URL `https://doi.org/10.1109/ICCV48922.2021.00076`.

Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *international conference on machine learning*, pp. 1–9. PMLR, 2015.

Linchao Zhu, Sercan O Arik, Yi Yang, and Tomas Pfister. Learning to transfer learn: Reinforcement learning-based selection for adaptive transfer learning. In *European Conference on Computer Vision*, pp. 342–358. Springer, 2020.

Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. In *European conference on computer vision*, pp. 566–583. Springer, 2020.

A Common Settings in Transfer Learning

The transfer learning literature defines various settings to leverage training data from a different task and/or distribution. Although not all papers use the same definitions, Table 11 presents the most common settings with reference to the literature supporting these definitions.

Table 11: Classical Transfer Learning Settings

Transfer Learning leverages a source distribution in order to perform better on a target distribution (Thrun & Pratt, 1998).

Multitask Learning improves generalization by leveraging the information contained in the training signals of related tasks (Caruana, 1993; 1997).

Domain Adaptation aims to improve accuracy on target distribution with insufficient labeled data by leveraging a model trained on a different but related source distribution (Farahani et al., 2021).

Unsupervised Domain Adaptation considers the setting where labeled source domain data (x, y) are available for training, while only unlabeled (x) data from the target domain are available (Ganin & Lempitsky, 2015).

Distribution Shift considers that the test distribution is different from the training distribution, usually in the context where the model cannot be retrained to adapt to the new test conditions (Koh et al., 2021).

Gradual Distribution Shift is an online setting where the training distribution progressively evolves (Kumar et al., 2020).

Covariate shift corresponds to a predictive setting where the distribution over the input features p(x) is different at training and test time, while the posterior distribution p(y|x) does not change (Bickel et al., 2009).

Label shift corresponds to a predictive setting where the class prior p(y) between train and test changes but the conditional distribution p(y|x) is assumed identical (Garg et al., 2020).

Fine-tuning is a specific domain adaptation technique which considers training a model on the target domain from an initial model trained on the source domain (Matic et al., 1993; Denevi et al., 2018; Zhang et al., 2021).

Zero-Shot Task Transfer addresses new tasks at test time without updating the model (Larochelle et al., 2008). It typically relies on a way to represent novel tasks in order to condition the model, e.g. text prompts (Radford et al., 2019; Srivastava et al., 2022).

Few-Shot Task Transfer is similar to zero-shot transfer and does not update the model weights. As a difference, the task conditioning information provides few training instances along with the description of the tasks (Radford et al., 2019; Srivastava et al., 2022).

B Multi-Task Extension

We extend the proposed framework to the multi-task setting, when there are several downstream specific tasks for which one seeks a good model. We treat the outer loss as the *average loss* over the specific sets: let $D^1_{\text{specific}}, \ldots, D^T_{\text{specific}}$ a set of T specific sets. In Equation 2, we simply use

$$\mathcal{L}_{\text{specific}}(\theta^*(\alpha)) = \sum_{t=1}^{T} \sum_{x' \in D^t_{\text{specific}}} \ell(x'; \theta^*(\alpha)) \tag{5}$$

so that the outer loss minimization leads to good performance on average over the specific sets. Algorithm 1 is adapted to this case by sampling a specific batch B_{specific} at random between each specific set; the rest

Table 12: Multi-task experiment. “Dialogue” consists of the three sets `opensubtitles`, `openwebtext2`, and `stackexchange`, “Academic” consists of the sets `arxiv`, `pubmed`, and `wikipedia`.

Method	“Dialogue”			
	Avg.	`opensubtitles`	`openwebtext2`	`stackexchange`
Baseline	1.157	1.158	1.201	1.113
Classifier	1.148	1.167	1.154	1.122
CDS	② 1.140	1.180	1.163	1.077
SOBA	① 1.129	1.144	1.204	1.039
	“Academic”			
	Avg.	`arxiv`	`pubmed`	`wikipedia`
Baseline	0.966	0.994	0.942	0.960
Classifier	② 0.896	0.980	0.815	0.892
CDS	① 0.894	0.947	0.807	0.929
SOBA	0.908	0.945	0.848	0.931

of the algorithm is identical. We conduct such an experiment in the Language Modelling setting described in subsection 5.2, where the specific sets consist of $T = 3$ datasets from the Pile dataset. We consider two different subsets of 3 specific sets: “dialogue” sets {`opensubtitles, openwebtext2, stackexchange`} and “academic” sets {`arxiv, pubmed, wikipedia`}. In Table 12, we report results after fine-tuning on the specific loss — that is, models are fine-tuned on a mixture of the $T = 3$ datasets. We observe that the bilevel methods and the classifier both improve over the baseline. As expected, the resulting models are good on each dataset but are not as good as models pre-trained and fine-tuned on one dataset only, as shown in subsection 5.2.

C Hyper-parameters

Table 13: Hyperparameters for the LM experiment

Hyperparameter	Value
`batch_size`	128
`dropout_rate`	0.1
`big_batch_size`	16384
`optimizer`	adam
`learning_rate`	0.002
`meta_learning_rate`	0.001
`meta_optimizer`	adam
`num_steps`	300000

Table 14: Hyperparameters for the translation experiment

Hyperparameter	Value
`batch_size`	256
`dropout_rate`	0.1
`big_batch_size`	2048
`optimizer`	adam
`learning_rate`	0.0002
`meta_learning_rate`	0.001
`meta_optimizer`	adam
`num_steps`	500000

Table 15: Hyperparameters for the vision experiment

Hyperparameter	Value
`batch_size`	256
`big_batch_size`	2048
`optimizer`	SGD + momentum
`learning_rate`	0.05
`meta_learning_rate`	0.0001
`meta_optimizer`	adam
`num_steps`	500000