DoGE: Domain Reweighting with Generalization Estimation

Simin Fan
EPFL
simin.fan@epfl.ch

Matteo Pagliardini
EPFL
matteo.pagliardini@epfl.ch

Martin Jaggi
EPFL
martin.jaggi@epfl.ch

Abstract

The coverage and composition of the pretraining data corpus significantly impacts the generalization ability of large language models. Conventionally, the pretraining corpus is composed of various source domains (e.g. CommonCrawl, Wikipedia, Github etc.) according to certain sampling probabilities (*domain weights*). However, current methods lack a principled way to optimize domain weights for ultimate goal for generalization. We propose DOmain reweighting with Generalization Estimation (DoGE), where we reweigh the sampling probability from each domain based on its contribution to the final generalization objective assessed by a gradient-based generalization estimation function. First, we train a small-scale proxy model with a min-max optimization to obtain the reweighted *domain weights*. At each step, the domain weights are updated to maximize the overall generalization gain by mirror descent. Finally we use the obtained *domain weights* to train a larger scale full-size language model. On SlimPajama-6B dataset, with universal generalization objective, DoGE achieves better average perplexity and zero-shot reasoning accuracy. On out-of-domain generalization tasks, DoGE reduces perplexity on the target domain by a large margin. We further apply a parameter-selection scheme which improves the efficiency of generalization estimation.

1 Introduction

Pretrained Large Language Models (LLMs) demonstrate impressive generalization abilities, making them the workhorse of today's NLP research and many practical use cases (Devlin et al., 2019; Brown et al., 2020; Chowdhery et al., 2022; Touvron et al., 2023a;b). They are trained on very large text corpora collected from various source domains and can be quickly fine-tuned to many specific downstream tasks. The composition of a pretraining corpus often depend on the accessibility of each data sources, while not necessarily the optimal for training the model. For example, 72.56% tokens of RedPajama (Together Computer, 2023) are sampled from CommonCrawl, while only 4.65% are from GitHub. While recent research has demonstrated that the quantity and quality of the pre-training corpus could substantially affect model's effectiveness (Kaplan et al., 2020; Hoffmann et al., 2022; Longpre et al., 2023), there are in contrast relatively few explorations into how its composition from various source domains could contribute to the generalization ability of the language model (Lee et al., 2023; Hashimoto, 2021; Xie et al., 2023a). The *domain weights* adopted by current state-of-the-art LLMs are mostly determined by heuristics (Gao et al., 2020) or tuned according to a series of downstream tasks (Du et al., 2022), which can be sub-optimal and costly.

Recently, Xie et al. (2023a) proposed a learnability-based domain reweighting framework DoReMi, which settles *domain weights* using two small-scale auxiliary models: first, a reference model is "well-trained" using uniform domain weights; next, a second auxiliary model—referred to as proxy

model—is trained from scratch with the objective to find domain weights that minimize the worst-case *excess loss*, i.e. the per-domain loss gap between the proxy model and the well-trained reference model. They interpret the *excess loss* as an estimation for the remaining learnability of a given domain at each training step—a large gap indicating the proxy model can further learn to model the associated domain. Despite the encouraging empirical results of DoReMi, optimizing *domain weights* to maximize the overall learnability summed across all domains (i) creates a strong dependency on the well-trained model whose capacity can strongly influence the overall accuracy and requires appropriate tuning, and (ii) creates a dissonance between the ideal goal of minimizing the average validation loss across domains and the employed objective which seeks to simply mimic the well-trained model.

To mitigate these issues, we propose DOmain reweighting with Generalization Estimation (DoGE), which finds optimal *domain weights* by explicitly optimizing for best generalization to a given set of domains. We follow the two-stage process of DoReMi which consists of first obtaining domain weights using proxy models, and, in the second stage, using those weights to train a final larger model. In contrast to DoReMi, our objective does *not* require any well-trained model, instead allowing us to only rely on a single small-scale auxiliary model. At timestep t of the first stage when training the proxy model, we look for the mixture of domains that will greedily minimize the average loss across all domains at the next step $t + 1$. As we derive how to update the domain weights to achieve that goal, we find a reformulation of the problem with an intuitive interpretation and links to influence functions (Pruthi et al., 2020). In essence, our method can be summarized as follow:

> *A data domain should receive a large weight if (i) it contributes to the learning of other domains or (ii) if it is difficult to learn.*

During the optimization of the proxy model, we also leverage recent insights on influence function (Yeh et al., 2022) to improve our method by filtering out parameters with large *cancellation effect*. This has the additional effect of improving the computational complexity of DoGE. Finally, the domain weights averaged over all steps are taken as the domain-wise sampling weights to train a full-size language model. A visual overview of the DoGE method is shown in Fig. 1.

Contributions. We summarize our contributions as follows:

- We introduce and rigorously derive DoGE from the explicit objective of generalizing to a specific set of domains.
- We empirically show how our method outperforms strong baselines including DoReMi in terms of (i) average perplexity on the target domains, and (ii) reasoning capabilities.

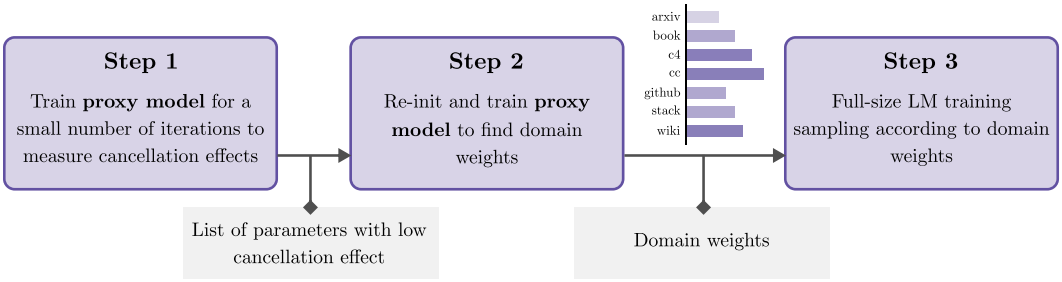


Figure 1: **Summary of DoGE** 🐶**.** Our method consists of three steps. **Step 1** identifies the parameters of the proxy model with low cancellation effect. Those parameters are more reliable to use in influence estimation based on gradients (Yeh et al., 2022). **Step 2** is the heart of our method. We learn domain weights which maximize the generalization of the proxy model to a pre-defined mixture of domains. This mixture can be the training data mixture D_{train} or containing domains absent from D_{train}. Finally, in **Step 3**, we use the computed domain weights to train a large LM.

2 Domain Reweighting with Generalization Estimation

In this section, we motivate and derive DoGE, for the goal of re-weighting training domains $D_{train} \triangleq \{D_1, \ldots, D_k\}$ to improve the model's generalization to a given set of domains. We distin-

guish two scenarios for generalization: (1) **Universal generalization**, where the ultimate objective is to minimize the validation loss across all training domains D_1 to D_k; as well as (2) **Out-of-domain generalization** where we aim at minimizing the validation loss on a specific D_{ood} domain, with $D_{ood} \notin D_{train}$. The later case is especially relevant when considering generalization to specific domains which are too small to have a significant impact when used during pretraining.

Setup & notation. Let $D_{train} \triangleq \{D_1, \dots, D_k\}$ be a large corpus split into k domains according to meta-attributes (e.g. source, topic). We aim to find ***domain weights*** $\boldsymbol{\alpha} \in \Delta^k \subset \mathbb{R}^k$ as a probability distribution over the probability simplex. The final data mixture used to train the full-size language model is constructed by first sampling a domain according to the domain-wise distribution $\boldsymbol{\alpha}$, followed by uniformly sampling an instance of that domain. This leads to the sample-wise distribution $P_{\boldsymbol{\alpha}} \triangleq \sum_{i=1}^{k} \alpha_i \cdot \text{unif}(D_i)$. In the following, we will describe how to optimize $\boldsymbol{\alpha}$ guided by training a proxy model of parameters $\boldsymbol{\theta}$ on D_{train}. For a domain D_i we denote by $l_i(\boldsymbol{\theta})$ the next token prediction loss of the proxy model on a minibatch sampled from domain D_i. Let $\bar{l}(\boldsymbol{\theta}) \triangleq \frac{1}{k} \sum_{i \in [k]} l_i(\boldsymbol{\theta})$ be the average loss. Let $|D|$ refer to the number of samples in D.

Universal generalization. In the case of universal generalization, our goal is to minimize $\bar{l}(\boldsymbol{\theta})$. This posit that all k given training domains have the same importance. As a point of comparison, note that the classical loss used to train large language models is $\tilde{l}(\boldsymbol{\theta}) = \sum_{i \in [k]} \frac{|D_i|}{|D_{train}|} l_i(\boldsymbol{\theta})$ which favors larger domains. One naive approach could consist in re-weighting samples by the inverse of the sampling probability: $\bar{l}(\boldsymbol{\theta}) = \sum_{i \in [k]} \tilde{\alpha}_i \frac{|D_i|}{|D_{train}|} l_i(\boldsymbol{\theta})$ with $\tilde{\alpha}_i = \frac{|D_{train}|}{|D_i|}$, however, this approach ignores everything of the complex interactions between the model weights $\boldsymbol{\theta}$ and the nature of the domains which can (i) overlap and (ii) be more or less challenging to learn. In practice, this naive approach does not fare well when compared to other methods (see § 3).

We instead propose to tie the learning of domain weights $\boldsymbol{\alpha} \in \Delta^k$ to the tuning of the proxy model parameters $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{(t+1)} \triangleq \boldsymbol{\theta}^{(t)} - \eta^{(t)} \sum_{i \in [k]} \alpha_i^{(t)} \nabla l_i(\boldsymbol{\theta}^{(t)}), \quad \text{with } \boldsymbol{\alpha}^{(t)} \in \Delta^k, \tag{1}$$

where $\eta^{(t)}$ is the step size and $\nabla l_i(\boldsymbol{\theta}^{(t)})$ is the stochastic gradient for samples of D_i at time-step t. This yields a bi-level optimization problem where the weights $\boldsymbol{\theta}^{(t)}$ are updated according to the training update (1) and $\boldsymbol{\alpha}^{(t)}$ is updated for a fixed $\boldsymbol{\theta}^{(t)}$. The update of the domain weights $\boldsymbol{\alpha}^{(t)}$ can be derived as follows. At time-step t, we can write our objective as follow:

$$\begin{aligned}
\boldsymbol{\alpha}_\star^{(t)} = \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min}\ \bar{l}(\boldsymbol{\theta}^{(t+1)}) &= \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min} \sum_{i \in [k]} [l_i(\boldsymbol{\theta}^{(t+1)}) - l_i(\boldsymbol{\theta}^{(t)})] \\
&\approx \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min} \sum_{i \in [k]} \langle \nabla l_i(\boldsymbol{\theta}^{(t)}), \boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)} \rangle \\
&= \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min} \sum_{i \in [k]} \langle \nabla l_i(\boldsymbol{\theta}^{(t)}), -\eta^{(t)} \sum_{j \in [k]} \alpha_j \nabla l_j(\boldsymbol{\theta}^{(t)}) \rangle
\end{aligned} \tag{2}$$

Let $W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i \in [k]} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle$ be the *generalization estimation function* on the j^{th} domain. Intuitively, this quantity measures the alignment of the learning tasks across domains: a high $W_j^{(t)}$ means learning D_j will also contribute to learning other domains. We write $\mathcal{W}^{(t)} = [W_1^{(t)}, \dots, W_k^{(t)}] \in \mathbb{R}^k$ for the score vector regrouping all generalization estimation functions. We can rewrite (2) simply as:

$$\boldsymbol{\alpha}_\star^{(t)} = \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min} -\eta^{(t)} \boldsymbol{\alpha}^\top \mathcal{W}^{(t)} \tag{3}$$

While a closed-form solution exist for $\boldsymbol{\alpha}_\star^{(t)}$, we find that it is too sensitive to noisy gradients. For the sake of stability, we instead only partially solve (3) using a single step of mirror descent (Nemirovski & Yudin, 1983; Beck & Teboulle, 2003) with a Bregman divergence $D_\Psi(\boldsymbol{\alpha} \| \boldsymbol{\alpha}^{(t-1)})$ with $\Psi(\boldsymbol{\alpha}) = \sum_i \alpha_i \log(\alpha_i)$:

$$\boldsymbol{\alpha}^{(t)} = \underset{\boldsymbol{\alpha} \in \Delta^k}{\arg\min} -\eta^{(t)} \boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + \mu D_\Psi(\boldsymbol{\alpha} \| \boldsymbol{\alpha}^{(t-1)}) \tag{4}$$

This yields the following multiplicative weights update rule, see e.g. (Beck & Teboulle, 2003):

$$\boldsymbol{\alpha}^{(t)} = \frac{\hat{\boldsymbol{\alpha}}^{(t)}}{\sum_{i\in[k]} \hat{\alpha}_i^{(t)}}, \quad \text{with} \quad \hat{\boldsymbol{\alpha}}^{(t)} = \boldsymbol{\alpha}^{(t-1)} \odot \exp\left(\frac{\eta^{(t)}\mathcal{W}^{(t)}}{\mu}\right) \tag{5}$$

The final algorithm is summarized in Alg. 1, at each time-step t, we alternate updating $\boldsymbol{\alpha}^{(t)}$ and $\boldsymbol{\theta}^{(t)}$.

Algorithm 1 DoGE Domain Reweighting (for Universal Generalization).

Input: Domain data splits $D_1, \ldots, D_k$, Proxy model weights $\boldsymbol{\theta}^{(0)}$, Hyperparameters: number of training steps T, batch size b, step size $\eta^{(t)}$, Bregman coefficient μ.
Initialize proxy weights $\boldsymbol{\theta}^{(0)}$
Initialize proxy domain weights $\boldsymbol{\alpha}^{(0)} = \frac{1}{k}\mathbf{1}$
for $t \in [T]$ **do**
  Uniformly sample minibatch $B^{(t)} = \{B_1^{(t)}, \ldots, B_k^{(t)}\}$ of size b
  Obtain $\nabla l_i(\boldsymbol{\theta}^{(t)}, B_i^{(t)})$ for $i \in [k]$ $\longrightarrow$ (***Forward/Backward pass***)
  Compute $\mathcal{W}^{(t)}$. Update domain weights according to Eq. (5):
    $\hat{\boldsymbol{\alpha}}^{(t)} \leftarrow \boldsymbol{\alpha}^{(t-1)} \odot \exp(\eta^{(t)}\mathcal{W}^{(t)}/\mu)$
    $\boldsymbol{\alpha}^{(t)} \leftarrow \hat{\boldsymbol{\alpha}}^{(t)} / \sum_{i=1}^{k} \hat{\alpha}_i^{(t)}$
  Update $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta^{(t)} \sum_{i\in[k]} \alpha_i^{(t)} \nabla l_i(\boldsymbol{\theta}^{(t)}, B_i^{(t)})$
Return Domain weights $\bar{\boldsymbol{\alpha}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\alpha}^{(t)}$

Out-of-domain generalization. In the out-of-domain generalization scenario we want to generalize to a D_{ood} domain that is not part of D_{train}. In that instance the above derivation still holds with only minor modifications: (i) we are now considering our objective to be $l_{ood}(\boldsymbol{\theta})$ instead of $\bar{l}(\boldsymbol{\theta})$, and (ii) we now have $W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \nabla l_{ood}(\boldsymbol{\theta}^{(t)}) \rangle$, for clarity we call $\mathcal{W}_{ood}^{(t)} = [W_1^{(t)}, \ldots, W_k^{(t)}]$. The update of $\boldsymbol{\alpha}^{(t)}$ is the same as in (5) replacing $\mathcal{W}^{(t)}$ with $\mathcal{W}_{ood}^{(t)}$. The associated algorithm can be seen in Alg. 2, where all differences with Alg. 1 are colored in blue.

Algorithm 2 DoGE Domain Reweighting (for Out-of-domain Generalization).

Input: Training domain data splits $D_1, \ldots, D_k$, OOD domain D_{ood}, Proxy model weights $\boldsymbol{\theta}^{(0)}$, Hyperparameters: number of training steps T, batch size b, step size $\eta^{(t)}$, Bregman coefficient μ.
Initialize proxy weights $\boldsymbol{\theta}^{(0)}$
Initialize proxy domain weights $\boldsymbol{\alpha}^{(0)} = \frac{1}{k}\mathbf{1}$
for $t \in [T]$ **do**
  Uniformly sample minibatch $B^{(t)} = \{B_1^{(t)}, \ldots, B_k^{(t)}\} \cup \{B_{ood}^{(t)}\}$ of size b
  Obtain $\nabla l_i(\boldsymbol{\theta}^{(t)}, B_i^{(t)})$ for $i \in [k]$ and $\nabla l_{ood}(\boldsymbol{\theta}^{(t)}, B_{ood}^{(t)})$ $\longrightarrow$ (***Forward/Backward pass***)
  Compute $\mathcal{W}_{ood}^{(t)}$. Update domain weights according to Eq. (5):
    $\hat{\boldsymbol{\alpha}}^{(t)} \leftarrow \boldsymbol{\alpha}^{(t-1)} \odot \exp(\eta^{(t)}\mathcal{W}_{ood}^{(t)}/\mu)$
    $\boldsymbol{\alpha}^{(t)} \leftarrow \hat{\boldsymbol{\alpha}}^{(t)} / \sum_{i=1}^{k} \hat{\alpha}_i^{(t)}$
  Update $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta^{(t)} \sum_{i\in[k]} \alpha_i^{(t)} \nabla l_i(\boldsymbol{\theta}^{(t)}, B_i^{(t)})$
Return Domain weights $\bar{\boldsymbol{\alpha}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{\alpha}^{(t)}$

Link between $\mathcal{W}^{(t)}$ and influence functions. Following Pruthi et al. (2020), given samples from a source and target domain $B_s \sim D_s$ and $B_t \sim D_t$, the influence of D_s on D_t can be estimated by

$\mathcal{I}(B_s, B_t) = \langle \nabla l_s(\boldsymbol{\theta}), \nabla l_t(\boldsymbol{\theta}) \rangle$. Now observing the definition of $W_j^{(t)}$:

$$W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i \in [k]} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle = \underbrace{\langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i \in [k], i \neq j} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle}_{\text{out-of-domain influence}} + \|\nabla l_j(\boldsymbol{\theta}^{(t)})\|_2^2 \quad (6)$$

Thus the first term in (6) estimates the sum of influences from all the other $k-1$ domains on the j^{th} domain, while the second term denotes the magnitude of the gradient from domain D_j. Intuitively, a domain should be up-weighted when (i) it contributes to the learning of other domains (high out-of-domain influence), or (ii) when the domain itself has not been learnt enough (high magnitude of gradient for this domain). Those two mechanisms are precisely what Equ. (3) expresses.

Cancellation effect. Yeh et al. (2022) have shown how a *cancellation effect* can diminish the discriminative power of gradient-based influence scores. The cancellation effect refers to parameters of the models not evolving much during training—the gradients they receive over the training process cancel each other—while nonetheless receiving large gradients on individual samples. Those parameters have little influence when considering large volumes of data, but can poison the influence scores when computed over small minibatches. Thus, computing $\mathcal{W}^{(t)}$ with the entire gradient risks being dominated by these malicious large gradients and lose its distinguish power. Following Yeh et al. (2022), we quantify the cancellation effect of each parameter, and drop those parameters from the gradient when they have a high cancellation effect. To measure the cancellation effect, we train another model—identical to the proxy model—on a small number of steps (e.g. 1000), and evaluate the cancellation effect for each module (e.g. linear, layer norm, etc.).

Summary of DoGE. We hereby summarize the three steps of our method, as seen in Fig. 1:

1. We train a proxy model on a small number of steps to measure the cancellation coefficient of each parameter or layer.
2. We re-initialize the proxy model and and train on D_{train} following Alg. 1 or Alg. 2. When computing $\mathcal{W}^{(t)}$ or $\mathcal{W}_{ood}^{(t)}$, we only consider parts of the gradients with a low cancellation effect. We obtain a domain weight distribution $\bar{\boldsymbol{\alpha}} \in \Delta^k$.
3. Finally we train the full size model by sampling D_{train} according to $P_{\bar{\boldsymbol{\alpha}}} \triangleq \sum_{i=1}^{k} \bar{\alpha}_i \cdot \text{unif}(D_i)$

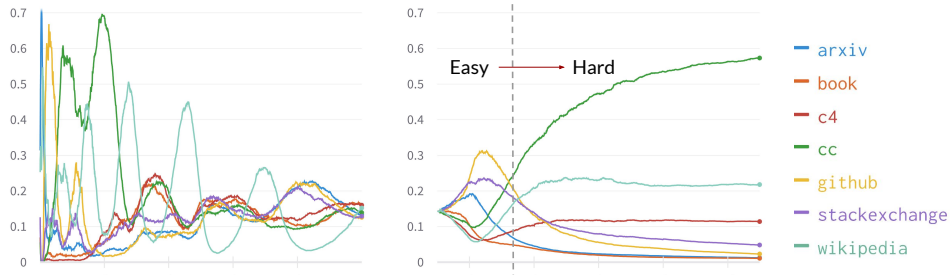
3 DOGE Improves Generalization Ability

In this section, we showcase how DoGE improves the model's performances in both the universal generalization and domain-specific generalization settings. We compare domain weights returned by DoGE and DoReMi as well as their evolution during the training of the proxy model. In the case of universal generalization, we show how DoGE can outperform other methods on downstream tasks as well as in terms of average perplexity. In the case of out-of-domain generalization, we visualize domain weights from DoGE and observe a significant improvement in perplexity on the D_{ood} domain.

Training setup. We experiment on a 6B subset[1] from SlimPajama (Soboleva et al., 2023), which include 7 domains. We train a small 82M decoder-only transformer (Vaswani et al., 2023) LM as the proxy model for domain reweighting. The updated domain weights are further applied to train the base model of 124M parameters. All models are trained from scratch with batch size of 70, and maximum token length of 512. The vocabulary size of the tokenizer is 50304.

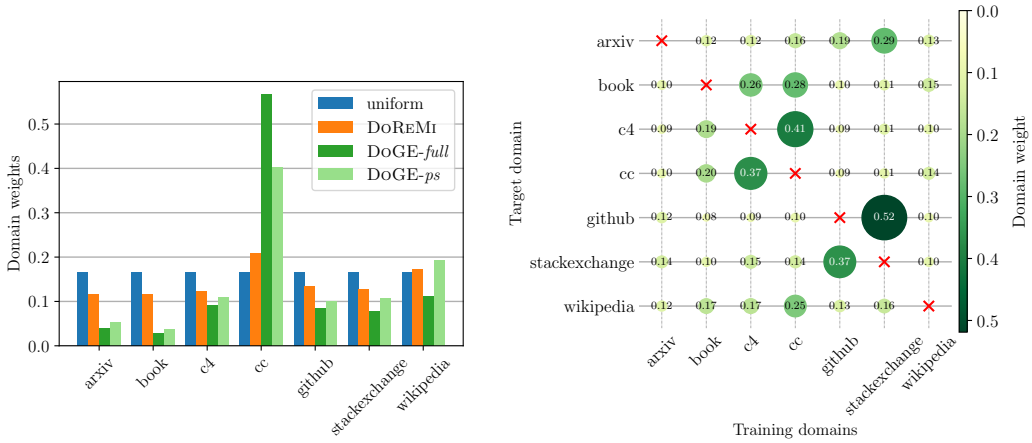
DoGE-*full* & DoGE-*ps*. To show the impact of filtering parameters according to the cancellation effect, we consider two version of DoGE. DoGE-*full* computes the generalization estimation score using the full model gradients, without any filtering, while DoGE-*ps* (parameter selection) only selects the 10 parameter matrices (corresponding to 20.9% of the full gradient) with the lowest cancellation effect when updating $\boldsymbol{\alpha}$. When no specification is given DoGE refers to DoGE-*ps*.

[1] https://huggingface.co/datasets/DKYoon/SlimPajama-6B



(a) DoReMi (82M) (b) DoGE (82M)

Figure 2: **Domain weights evolution when training a (82M parameters) proxy model for DoReMi (left) and DoGE-*ps* (right).** In **(a)**, the domain weights from DoReMi oscillate greatly during training while tend to converge to uniform in the end. In contrast, DoGE **(b)** shows clear phase transitions during the training process: in earlier stage, DoGE upweights easier domains (Github and Stackexchange) while up-weighing CC and Wikipedia in the late stage, which is considered more diverse and harder to learn.



(a) Universal Generalization (b) Out-of-domain Generalization

Figure 3: **Domain weights for universal and out-of-domain generalization.** In **(a)**, we compare uniform domain weights with those obtained by DoReMi and DoGE. We see how DoGE seem to upweights significantly the common-crawl (cc) domain. In **(b)** we show DoGE domain weights for out-of-domain generalization. Those domain weights obtained by training on various training mixtures consisting of all the training domains except for one which is used as target (D_{ood}) domain. Each row represents a distribution returned by Alg. 2. The target domain is not used during training and hence is marked by a red cross. The weight distributions look very coherent, e.g. to generalize to GitHub, DoGE upweights stackexchange which contains a significant fraction of code. Similarly, to generalize to cc, the c4 domain—which also consists in web data—is upweighted. The proxy models used posses 82M parameters.

3.1 Universal Generalization

In the case of universal generalization, we aim to improve the model's generalization across all domains present in the training set. We measure the average perplexity across all domains and the exact match accuracy on the Lambada word prediction task Paperno et al. (2016). Moreover, we probe the reasoning abilities of the various models on a series of zero-shot reasoning tasks: SciQ (Welbl et al., 2017), WiC (Pilehvar & Camacho-Collados, 2019), WinoGrande (Sakaguchi et al., 2019), OpenbookQA (Mihaylov et al., 2018) and COPA (Gordon et al., 2012). We compare our proposed algorithm to (i) a random baseline with uniform domain weights; and (ii) DoReMi (Xie et al.,

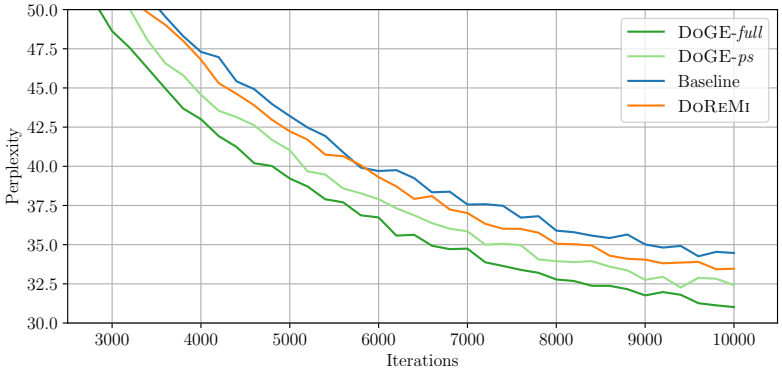


Figure 4: **Average perplexity across domains when training the final model.** For both DoReMi and DoGE, we first obtain domain weights using a 82M proxy model, and subsequently use those domain weights to train a 124M base model (82M $\rightarrow$ 124M). The "baseline" model consists in a 124M model trained using uniform domain weights. DoGE-*full* computes the generalization estimation function $\mathcal{W}^{(t)}$ using the gradients w.r.t. all the parameters, while DoGE-*ps* uses only a subset of 10 parameter matrices selected according to their cancellation effect (see § 2). Both DoGE-*ps* and DoGE-*full* outperform the uniform baseline and DoReMi.

2023a). For the later, we use the same batch size, context length and vocabulary size as for DoGE, while taking all other DoReMi-specific parameters from (Xie et al., 2023a).

Evolution of domain weights. Figure 2 shows the domain weights evolution during the training of the proxy model. When applying DoReMi, the domain weights change drastically at the beginning of training, while gradually converging to uniform along the training. In comparison, DoGE shows a clear phase transition in different stages of training: in the early stage, DoGE upweights GitHub and Stackexchange while gradually upweighing CC and Wikipedia in the later phase of training. This aligns with the intuition of *curriculum learning* that the model would first learn from *easy knowledge* (Github/Stackexchange) while moving on to more challenging and diverse ones (CC/Wikipedia) in a later phase.

Final domain weights used. In Fig. 3.a we show the final domain weights adopted by different methods. For DoGE and DoReMi, those are obtained by averaging the $\boldsymbol{\alpha}$ distributions obtained while training the proxy model (see returned $\bar{\boldsymbol{\alpha}}$ in Alg. 1 and Alg. 2).

Assessment on language modeling and reasoning ability. In both Fig. 4 and Table 1, we see DoGE's average perplexity on the held-out set is better than those of baseline methods. DoGE-full improves the average domain perplexity the most, while both DoGE-ps and DoGE-full outperform both the uniform baseline and DoReMi. This demonstrates our method's ability to better learn the training domains D_{train}. Moreover, accuracies on reasoning tasks in Tab.2 as well as on Lambada in Tab. 1 show DoGE outperforming the baselines.

Table 1: **Universal generalization results.** We compare DoGE with DoReMi and uniform sampling of domains on three metrics: (i) the average perplexity across domains showing how well different methods fit the training domains, (ii) the Lambada and exact match average accuracies on reasoning tasks. DoGE outperforms other methods on those three metrics. The detailed scores used in the computation of the average reasoning accuracy are shown in Table 2.

	Avg. pplx (↓)	Lambada Acc.[2] (↑)	Avg. Reasoning Acc. (↑)
Baseline (124M)	34.47	0.33	39.46
DoReMi (82M->124M)	33.47	0.41	38.40
DoGE-*full* (82M->124M)	**31.02**	**0.47**	39.24
DoGE-*ps* (82M->124M)	32.42	0.43	**40.00**

Table 2: **Exact-match accuracies for zero-shot reasoning tasks.** Overall DoGE improve downstream performance over the baselines.

	Baseline (124M)	DoReMi (82M->124M)	DoGE-*full* (82M->124M)	DoGE-*ps* (82M->124M)
SciQ	**25.40**	21.10	24.50	23.90
WiC	50.00	50.00	**50.47**	50.00
COPA	58.00	56.00	56.00	**61.00**
WinoGrande	50.51	50.28	51.22	**51.30**
OpenbookQA	13.40	**14.60**	14.00	13.80
Average	39.46	38.40	39.24	**40.00**

3.2 Out-of-Domain Generalization

Perplexities on domains outside of D_{train}. For out-of-domain generalization, we experiment with setting each of the domains in SlimPajama as the target domain, while using the remaining six domains as training set D_{train}. We run the proxy model (82M) for 10000 steps and then train the base model (82/124M) for 10000 steps. To improve efficiency, we select 10 parameter matrices with the least cancellation effect to compute the generalization estimation $\mathcal{W}_{ood}$. We evaluate the perplexity on the target domain on the heldout test set and show the results in Table 3.

Visualizing domain affinities. By setting each domains as the target domain, we can take a glance at the inter-domain affinities and dependencies through the domain weights of remaining training domains. Fig. 3.b shows how DoGE can up-weight domains close to D_{ood}. For instance, in order to generalize to D_{ood} = GitHub, DoGE naturally decides to give a large weight to the stackexchange domain, which is known to contain a lot of samples for various programming languages. With the domain weights from DoGE, ***DoGE's* 82*M model outperforms the* 124*M baseline model trained with uniform domain weights by a large margin*** across all target domains except for Wikipedia.

Table 3: **Out-of-Domain generalization results.** Perplexity (↓) on the target domain while training on the remaining 6 domains. DoGE significantly outperforms the uniform averaging baseline, demonstrating its ability to select which domain to upweight given its similarity to the target domain. Domain weights can be seen in Fig. 3.b.

	Baseline (82M)	DoGE (82M → 82M)	Baseline (124M)	DoGE (82M → 124M)
Arxiv	20.011	**18.578**	18.595	**17.351**
Book	81.129	**69.810**	75.042	**64.225**
C4	95.371	**77.900**	89.162	**72.015**
CommonCrawl	90.861	**75.156**	84.752	**69.517**
Github	7.334	**6.131**	6.956	**5.726**
StackExchange	18.622	**16.628**	17.407	**15.436**
Wikipedia	**50.866**	51.381	48.701	**46.889**

4 Discussion and Limitations

Poor adaptability to training phases. Recent research shows that language models demonstrate similar learning patterns as children, and acquire linguistic skills in a systematic order (Evanson et al., 2023). Thus, the language model requires distinct data mixtures in various training phases. While the domain weights evolution of DoGE (Fig. 2) seem to possess such learning phases—upweighing different domains at different times—a fixed set of domain weights is applied throughout the whole training process of the base model, which fails to adapt to different learning phases. How to detect the various phase transitions and design a more adaptive reweighting algorithm is an interesting future direction.

Vulnerability to the noisy domain. Our generalization estimation function is largely dependent on the gradient: a domain with a larger magnitude of gradient would have higher scores even when its alignment with other domains is poor. Thus, noisy domains are likely upweighed solely due to their high gradient magnitude.

In-domain variances and fine-grained clusters. Despite web-scraped data collection (e.g. CommonCrawl, C4) containing diverse textual content with various attributes, both DoGE and DoReMi assign a single weight on all samples within one domain. We believe applying reweighting on more fine-grained domains could help to detect in-domain variances and distinguish among subpopulations within one diverse domain.

5 Related Work

Data Selection for Language Modeling. Many works show how a rigorously selected training corpus can effectively improve downstream performance with fewer training tokens. On the CommonCrawl dataset, Marion et al. (2023) achieve better test-set perplexity by applying perplexity-based pruning, using only 30% of the tokens compared to the no-pruning baseline. Gunasekar et al. (2023) and Li et al. (2023) trained a 1.3B model Phi-1 using a corpus of 7B *text-book quality* code data, outperforming previous models trained on an approximately $15\times$ larger standard code dataset, illustrating the potential of high-quality data for code generation. Longpre et al. (2023) discover a trade-off between a model's toxic generalization behavior and its generalization ability by applying quality control with various thresholds.

Meanwhile, practitioners are exploring effective data selection methodologies to improve the training efficiency of LLM. However, due to the large size of pretraining datasets and the complexity of language modeling tasks, most traditional data selection methods fail to be applicable due to scalability issues. Classifier-based data filtering techniques are commonly used to construct a pretraining corpus (Gao et al., 2020; Penedo et al., 2023). Everaert & Potts (2023) propose GIO to select a training set that minimizes the KL-divergence to the target distribution, yet their experiments are limited to relatively small datasets ($\approx$ 10M tokens) due to high computation costs. By reducing the high-dimensional feature space into an n-gram-featured subspace, (Xie et al., 2023b) present a scalable importance resampling method which selects a subset aligning with a specific target distribution. Despite the improved scalability, the n-gram features may not be able to represent sophisticated semantics in the training corpus. Moreover, their methods require access to the downstream tasks as the target distribution, which is hard to apply in the general LM pretraining.

Domain Reweighting for LLM Pretraining. Compared to instance-wise data selection, domain reweighting aims to sample from various data groups. It is considered a more scalable method for language model pretraining. DoReMi(Xie et al., 2023a)—already described in § 1—is a good example of such methods. Chen et al. (2023) propose to build an online resampling curriculum by exploiting the dependency relationship among skills, where each domain can be considered as one skill. The relationship among N skills can be represented by a directed skill graph. However, constructing this ordered skill graph requires training of $\mathcal{O}(N^2)$ models, which reduces its applicability to general language model pretraining.

6 Conclusion

We introduced DoGE, a gradient-based domain reweighting framework, which finds the optimal domain weights tailored to various generalization objectives. Our experiments show DoGE is able to improve LLM's universal and out-of-domain generalization ability in terms of perplexity and zero-shot reasoning accuracy. Scaling-up experiments with larger models and datasets is an important future direction.

References

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Oper. Res. Lett.*, 31(3):167–175, 2003.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin,

Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

Mayee F. Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models, 2023.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. Glam: Efficient scaling of language models with mixture-of-experts, 2022.

Linnea Evanson, Yair Lakretz, and Jean-Rémi King. Language acquisition: do children and language models follow similar learning stages?, 2023.

Dante Everaert and Christopher Potts. Gio: Gradient information optimization for training dataset selection, 2023.

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, 2020.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021. URL https://doi.org/10.5281/zenodo.5371628.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pp. 394–398, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL https://aclanthology.org/S12-1052.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need, 2023.

Tatsunori Hashimoto. Model performance scaling with multiple data sources. In *International Conference on Machine Learning*, 2021. URL https://api.semanticscholar.org/CorpusID:235826265.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

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, 2020.

Alycia Lee, Brando Miranda, and Sanmi Koyejo. Beyond scale: the diversity coefficient as a data quality metric demonstrates llms are pre-trained on formally diverse data, 2023.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: phi-1.5 technical report, 2023.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, toxicity, 2023.

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, 2023.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018.

A. Nemirovski and D. Yudin. *Problem complexity and Method Efficiency in Optimization*, volume 1. Wiley, New York, 1983.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context, 2016.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Mohammad Taher Pilehvar and Jose Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations, 2019.

Garima Pruthi, Frederick Liu, Mukund Sundararajan, and Satyen Kale. Estimating training data influence by tracing gradient descent, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama, 2023. URL https://huggingface.co/datasets/cerebras/SlimPajama-627B.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413.

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, 2023a.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language models via importance resampling, 2023b.

Chih-Kuan Yeh, Ankur Taly, Mukund Sundararajan, Frederick Liu, and Pradeep Ravikumar. First is better than last for language data influence, 2022.