D4: Improving LLM Pretraining via Document De-Duplication and Diversification

Kushal Tirumala [*1] **Daniel Simig** [*1] **Armen Aghajanyan** [1] **Ari Morcos** [1]

Abstract

Over recent years, an increasing amount of compute and data has been poured into training large language models (LLMs), usually by doing one-pass learning on as many tokens as possible randomly selected from large-scale web corpora. While training on ever-larger portions of the internet leads to consistent performance improvements, the size of these improvements diminishes with scale, and there has been little work exploring the effect of data selection on pre-training and downstream performance beyond simple de-duplication methods such as MinHash. Here, we show that careful data selection (on top of de-duplicated data) via pre-trained model embeddings can speed up training (20% efficiency gains) and improves average downstream accuracy on 16 NLP tasks (up to 2%) at the 6.7B model scale. Furthermore, we show that repeating data intelligently consistently *outperforms* baseline training (while repeating random data performs worse than baseline training). Our results indicate that clever data selection can significantly improve LLM pre-training, calls into question the common practice of training for a single epoch on as much data as possible, and demonstrates a path to keep improving our models past the limits of randomly sampling web data.

1. Introduction

Due to computational limits, initial work on language model pre-training focused on training models on small, high-quality text datasets such as BookCorpus (Zhu et al., 2015) and Wikipedia (Merity et al., 2016). More recently, however, catalyzed by works like (Radford et al., 2019), advancements in large language models (LLMs) have been driven by leveraging large collections of unlabeled, uncurated data derived from snapshots of the internet (CommonCrawl (Raffel et al., 2020; Gao et al., 2020; Penedo et al.)), trading off small quantities of heavily-curated data for huge quantities of less-curated data. Because of the dramatic increase in data quantity, these strategies have resulted in higher performance models and have sparked a new paradigm wherein massive, largely unfiltered datasets are utilized for training (Chowdhery et al., 2022; Touvron et al., 2023; Smith et al., 2022).

Despite the essential role that large-scale web data now play in LM pre-training, data curation and selection for large-scale web data have not been thoroughly explored. This is primarily due to the universality of compute and data scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022) which give practitioners a low-risk way to reliably improve LM performance by merely adding "more" data, not necessarily the "right" data. Indeed, the data selection method used to model scaling laws (along with the data selection methods used in most LLM pre-training pipelines) involves simply randomly sampling tokens from web data dumps that have been put through a combination of simple heuristic filtering (e.g., to eliminate very short strings) and very near match de-duplication (Lee et al., 2021).

If we continue relying on scaling laws to improve LLMs, we will quickly hit diminishing returns due to the power-law nature of scaling laws. We will therefore need exponentially more data to maintain a consistent marginal improvement, which may prove especially challenging as we are fast approaching the limits of available human-generated text data (Villalobos et al., 2022). Encouragingly, in the context of vision, Sorscher et al. (2022) demonstrated that we could leverage simple data selection strategies to overcome costly power-law scaling. They compare numerous data selection methods and find that clustering data points in a pre-trained embedding space and ranking according to the distance to the cluster centroid ("SSL Prototypes") significantly improves the data efficiency of vision models. Recently, Abbas et al. (2023) demonstrated that using a pre-trained embedding space to de-duplicate data ("SemDeDup") improves both efficiency and performance of vision-language models

[*]Equal contribution [1]FAIR, Meta AI. Correspondence to: Kushal Tirumala <kushaltirumala99@gmail.com>, Daniel Simig <simigd@gmail.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

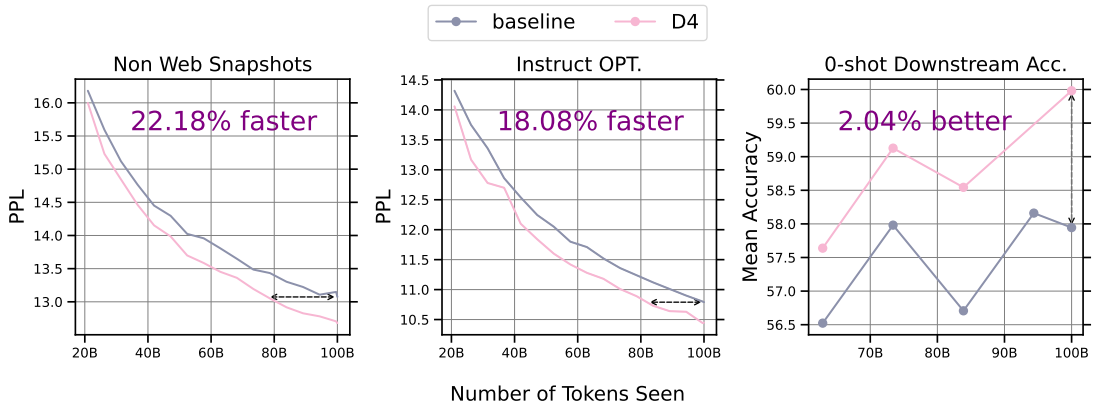

Figure 1. Learning curves for 6.7B OPT model pretraining on 100B tokens, with data selected with D4 (pink line) and randomly (gray line). D4 significantly outperforms baseline training, getting between 18-20% efficiency gains on validation perplexity and 2% increase in average 0-shot downstream accuracy across 16 NLP tasks. See Section A.2 for full learning curves.

such as CLIP. However, there has been little exploration of these or related approaches in training LLMs at scale. Motivated by this, we argue that by combining these approaches and applying them to LLMs, relatively simple data selection strategies leveraging pre-trained embeddings can significantly improve LLM training. Specifically, our contributions are as follows:

- We investigate different data selection strategies for standard LLM pre-training setups where data has already been manually filtered / de-duplicated (e.g., MinHash), and where we do not know the target distribution for which we optimize performance. We argue that the performance of current methods such as SSL Prototypes and SemDeDup are affected by duplicate-driven clusters in the embedding space. In Section 2.4 we propose a new data selection strategy **D4** to avoid getting impacted by such clusters.

- In Section 3.1, we show that in the *compute-limited regime* where we have "infinite" source data and train models with fixed token budgets, we can achieve better pre-training perplexity and downstream accuracy than random iid data selection and previously established methods. Furthermore, we show that our method D4 can achieve around 20% efficiency gains at the 6.7b model scale, and that the magnitude of efficiency gains increases with model scale.

- In the *data-limited regime*, where we run out of data and must epoch over data, cleverly choosing what data to repeat can beat training on randomly selected new data, whereas randomly choosing data to repeat underperforms adding new data (Section 3.2). This calls into question the standard practice of single epoch LLM training, and suggests that epoching over intelligently subselected data might be a better approach.

2. Experimental Setup

Notation Given a source dataset, D_{source}, of documents and model architecture, M, we aim to find a data selection strategy S that maximizes some evaluation metric $E(M(D_{S,R}))$. R indicates the proportion of remaining documents from the source dataset D_{source} after selecting data with strategy S. For this reason, we refer to R throughout this work as the *selection ratio*: for example, if $R = 0.25$ and $|D_{source}| = 100$ million, then we *select* 25% of documents from a source dataset of size 100M documents to arrive at a a training dataset with 25M documents. Throughout the paper, we use random selection as the baseline for S, as it is the most common method for selecting data for language model pre-training. In the rest of this section, we describe our choices of source dataset (D_{source}), model (M), evaluation metric (E), and, most importantly, our suggestions for the selection strategy (S).

2.1. Training Dataset (choice for D_{source})

We perform all of our training runs on a version of CommonCrawl pre-processed with a CCNet (Wenzek et al., 2019) pipeline identical to the one used by Touvron et al. (2023). We add an additional step of MinHash-based de-duplication (see more details in Section A.1). Applying this common step before our experiments guarantees that any effects observed in our experiments complement the currently prevalent approach of MinHash-based data de-duplication strategies. Throughout the rest of this work, we refer to this dataset as *CC-dedup*.

2.2. Model Training (choices for M and T_{target})

To evaluate different configurations of data selection strategies, we train OPT (Zhang et al., 2022) models from scratch on the pruned versions of datasets. We use the standard model architectures and settings of Zhang et al. (2022) and use MetaSeq (Zhang et al., 2022) to train all our models. For 125M models, we train to $T_{target} = 3B$ tokens. For 1.3B OPT models, we train to target token count of $T_{target} = 40B$. For 6.7B OPT models, we train to $T_{target} = 100B$ tokens. We choose these trimming down the token budgets suggested by Hoffmann et al. (2022) to meet our compute limitations. We provide full details of our training setup in Section A.1.

2.3. Evaluation Metrics (choices for E)

We keep most of our evaluation consistent with the setup from Zhang et al. (2022).

Validation Set Perplexity. Our validation sets mainly come from from (Zhang et al., 2022), which includes validation sets derived from subsets of the Pile (Gao et al., 2020) such as CommonCrawl, DM Mathematics, HackerNews, OpenSubtitles, OpenWebText2, Project Gutenberg, USPTO, Wikipedia, and PushShift.io Reddit (Baumgartner et al., 2020) (which we refer to as redditflattened). In addition, we measure perplexity on a validation set obtained from a train-validation split of our source dataset *CC-dedup*, and a validation set from C4 (Raffel et al., 2020).

We notice that the effects of data selection vary significantly on individual validation sets depending on whether the validation set was derived from a web data corpus or not (see more details and analysis in Section 3.4.1). Motivated by this, we split validation sets into Web-snapshots (C4, CommonCrawl, and CC-dedup) and Non-web snapshots, and report average perplexity within these sets.

Downstream Task Accuracy. To evaluate downstream performance of our trained models, we report average 0-shot accuracy across the 16 NLP tasks from Zhang et al. (2022), and use a prompting methodology consistent with Zhang et al. (2022). These set of 16 NLP tasks include Arc Challenge and ArcEasy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), StoryCloze (Mostafazadeh et al., 2016), Winograd (Levesque et al., 2012), Winogrande (Sakaguchi et al., 2021), as well as tasks from SuperGLUE (Wang et al., 2019). We refer the reader to Zhang et al. (2022) for more information about this evaluation setup.

Instruction Tuning Perplexity. The evaluation mentioned above metrics presents an inherent trade-off. Though accuracy on downstream tasks is typically viewed as a more concrete representation of a language model's real-world value, its variance tends to be higher due to the limited number of examples in these tasks and the step-wise behavior of accuracy as a metric. In contrast, perplexity, as a metric, is smoother while still exhibiting a strong correlation with performance (Schaeffer et al., 2023). Therefore as a middle ground between the two evaluation metrics, we propose evaluating the perplexity on a sample drawn from the instruction-tuning dataset used for fine-tuning OPT-IML (Iyer et al., 2022). This dataset spans over 1500 unique NLP tasks and comprises a wide array of prompt-answer pairs and therefore is representative of the *average* NLP task. It has been carefully crafted by merging extensive task collections such as Super-NaturalInstructions (Wang et al., 2022) and PromptSource (Bach et al., 2022). We refer the reader to Table 2.1 in (Iyer et al., 2022) for a comprehensive breakdown. This approach allows us to balance practical performance measures and statistical consistency in evaluation.

2.4. Data Selection Strategies (choices for S)

We focus our efforts on data selection strategies that use pre-trained model embeddings to select data due to their recent success in data pruning in vision and vision-language models (Abbas et al., 2023; Sorscher et al., 2022). We embed each document by feeding it into a 125M OPT model and use the last-layer embedding of the last token. All methods described below manipulate data points based on these embeddings.

SemDeDup: Abbas et al. (2023) proposed de-duplicating in both text and image domains by first using K-Means to cluster the embedding space, and removing points in each cluster that are within epsilon-balls of one another. We use this algorithm without any modifications and refer the reader to Abbas et al. (2023) for implementation details of this algorithm.

Prototypicality: Sorscher et al. (2022) investigated a large variety of data pruning strategies to improve the data efficiency of training image classification models, including a newly introduced "SSL Prototypes" metric that proved to be one of their best methods. This strategy involves first clustering the embedding space using k-means clustering and discarding data points in increasing order of their distance to the nearest cluster centroid, such that the most "prototypical" data points are discarded, enriching the much higher variance outliers. We refer the reader to Sorscher et al. (2022) for a more detailed description of this algorithm.

Both methods heavily rely on the quality of the clustering of the embedding space. Upon qualitatively analyzing our embedding space, we find many instances of duplicate-driven clusters: clusters of templated text or extremely semantically redundant information (see Section A.5 for examples) that were not removed by MinHash. These regions of embedding space tend to be very dense and cause k-means to

waste valuable cluster assignments on duplicated text. This biased clustering is also likely to impact the effectiveness of SSL Prototypes since many clusters will be entirely driven by duplicates. This insight leads us to the motivation behind our proposed strategy:

1. Apply *SemDeDup* with an overhead ratio R_{dedup}, producing a dataset D'
2. Re-cluster points in D' with K-Means
3. Apply *SSL Prototypes* on D', with an overhead ratio R_{proto}

The above-described strategy has an overall ratio of $R = R_{dedup} * R_{proto}$ and intends to diversify the distribution of our data locally and globally. For brevity we refer to this method as **D4**, a shorthand for *Document De-Duplication and Diversification*. Throughout this work, we choose $R_{dedup} = 0.75$ and vary R_{proto} (we discuss this choice in detail in Section A.1). In Section 3, we compare the performance of D4 to baseline training and other methods, and in Section 3.4 we analyze D4 and show that reclustering after semantic de-duplication indeed reduces the impact of duplicate-driven clusters (see Figure 7).

3. Results

3.1. Fixed compute regime: can data selection help on fixed token budgets?

In this section, we consider the fixed compute setting, where we curate and train on a fixed token budget by jointly increasing the size of the source dataset D_{source} and decreasing R (the fraction of the D_{source} which is selected), such that the target token budget remains constant. This setting is analogous to the most common paradigm for LLM training. As D_{source} grows and R decreases, we select from larger and larger initial datasets, resulting in a larger set of high-quality data points to select from and increasing the overall quality of the selected set. For clarity, we plot performance as a function of the ratio of the D_{source} to D_{target}. For each setting, we evaluate the performance of a baseline, SemDeDup alone, SSL Prototypes alone, and our proposed method D4.

Validation Perplexity. In Figure 2, we show that a relatively small amount of data selection using any of the three methods (small R) brings consistent improvements on all validation sets. However, as we increase R, we observe *opposing effects* on web snapshot and non-web-snapshots validation sets. We analyze this discrepancy in-depth in Section 3.4. However, on the Instruct OPT validation set, which corresponds much more closely to the the high-quality generations we want our LLMs to achieve, we found that all three methods led to consistent and clear perplexity improvements. Notably, we found that while all three methods provided benefits, D4 outperformed using both SemDeDup and SSL Prototypes independently, with the most notable gains exhibited when the source dataset is around 4x the target dataset size. Given that D4 consistently improves with source dataset size, we estimate this gap to grow with source dataset size.

Downstream Task Accuracy. In Figure 2, we also report 0-shot downstream accuracy averaged across a suite of NLP tasks. While the high variance of downstream accuracy makes it challenging to identify clear trends in the performance of various models, we again observe that 0-shot downstream accuracy generally increases with source dataset size.

Our findings also hold at larger model scales. We pick our best-performing configuration from 1.3B OPT experiments (e.g., $R = 0.25$) and train 6.7B OPT models on 100B tokens. Figure 1 shows the positive effects of applying D4 with $R = 0.25$ for a 6.7B model. The model trained on the pruned data reaches the same perplexity as the baseline model using 20% fewer update steps on average and achieves a 2% improvement in accuracy on our suite of downstream tasks at the end of the training - about as much difference as was reported by Zhang et al. (2022) between the OPT and GPT-3 family of models on the same set of tasks (See Figure 3 of Zhang et al. (2022)).

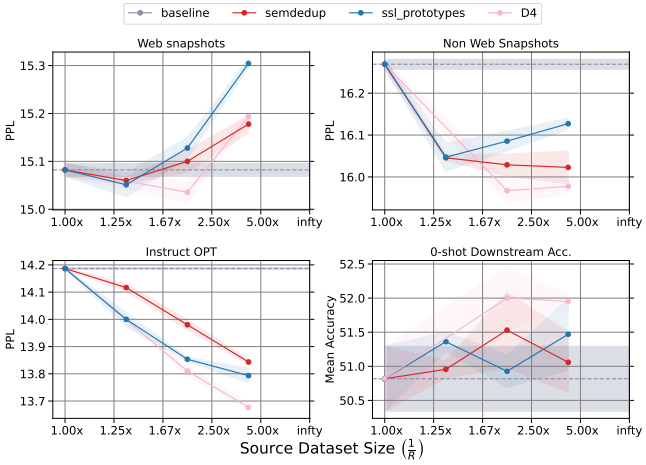

Figure 2. Comparison of data selection methods on validation perplexity. Each point denotes a 1.3B OPT model trained on 40B tokens. The x-axis denotes the size of the source dataset from which data is selected to achieve 40B tokens. The y-axis for the top 2 and right bottom graph depicts perplexity; the bottom left graph is average downstream on 16 NLP tasks from Zhang et al. (2022). The grey line denotes the value for baseline training. Shaded error is standard error across 3 seeds.

3.2. Fixed data regime: what happens when we run out of data?

The results in Section 3.1 indicate that, given a fixed amount of compute for training, selecting data from larger and larger source datasets is a promising method to improve language model performance. However, there is a practical limit to how much data can be curated from the web and, therefore, a natural limit to the size of the source dataset. What happens when we run out of data? Hernandez et al. (2022) found and analyzed disproportionately adverse effects of repeated data points in the training data. Similarly, concurrently to our work Muennighoff et al. (2023) shows that test loss deteriorates when epoching over a random subset of C4 more than four times. In this section, we investigate how the use of D4 affects model performance in this limited data, multi-epoch setting.

To test this, we assume a fixed token budget and a fixed data size which matches the token budget. We evaluate training on all the data as well as for two epochs on subsets of the data selected either randomly or using D4. We trained 1.3B parameter OPT models on these configurations and report average perplexity in Table 1. Unsurprisingly, epoching over a randomly selected subset of the data instead of using all the available data once leads to a slight degradation in model perplexity. In contrast, repeating data selected by D4 leads to an improvement in perplexity and downstream accuracy over randomly sampling new tokens. In other words, it is beneficial to select data via D4 and epoch 2 times, instead of doing one-pass learning on all available data. As seen in Figure 3, this finding generally holds across training as well. We refer to Section A.7 for results across model scale and data selection ratio.

To the best of our knowledge, this is the first result to demonstrate the benefits of repeating data for LLMs over randomly sampling new tokens via a principled data selection technique. We argue that the optimal way of using large-scale web data to pre-train LLMs could be: strategically choose a significantly smaller but better-distributed subset of the data and epoch over it multiple times.

3.3. Cost of data selection

In Section 3.1, we find that by training a 6.7B parameter model on data selected by D4, we reach the final perplexity of a baseline model using 20% fewer model updates. In our particular setup, this translates to **saving approximately 4300 GPU hours**. To demonstrate our method's practicality, we must ensure the cost of selecting data is significantly less than this. As described in Section 2.4, selecting data via D4 involves: first, embedding documents via a 125M OPT model; second, computing K-Means indices + distance to indices.

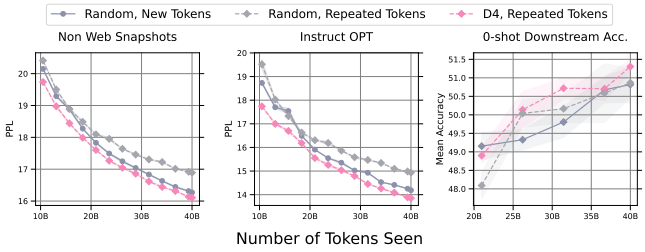

Figure 3. Comparing new tokens vs. repeated tokens for random data selection and D4 for fixed selection ratio $R = 0.25$. Each method chooses 25% of documents from the source dataset D_{source}, and epochs over that subset until the target token budget of 40B is reached. We observe that repeating tokens via D4 outperforms baseline training (random, new tokens).

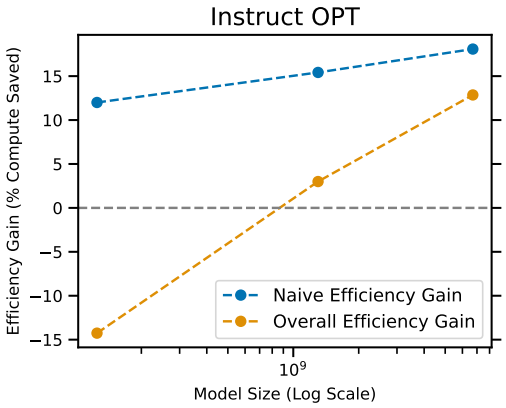

Figure 4. Efficiency gain as a function of model size on Instruct OPT perplexity. The blue line denotes the raw efficiency gain. The orange line denotes the overall efficiency gain, considering the compute necessary to apply D4 at $R = 0.25$.

The first step is completed on a single machine with 96 CPU cores in approximately one day. Given the two orders of magnitude difference between the prices of CPU and GPU cores [1], we consider this cost negligible. For the second step, embedding 400B tokens with a 125M parameter model takes approximately 888 GPU hours, using the same A100 GPUs. Comparing this with the 4300 GPU hours savings, we can conclude that D4 saves thousands of GPU-hours for models at the 6.7B scale. In Figure 4, we redo this calculation for different model sizes and we see that demonstrate that overall efficiency gain increases with model size. Based on this, we can conservatively [2] estimate that D4 would have overall efficiency gains of 20% for LLama-65B (Touvron

[1] Source: https://aws.amazon.com/ec2/pricing/on-demand/

[2] e.g. assuming 20% naive efficiency gains continue, although this is an underestimate since naive efficiency gains increase with model size as seen in Figure 4

S	T_{total}	$T_{selected}$	Epochs	Non-Web Snapshot PPL	Instruct PPL
Random	40B	40B	1	16.27 ± 0.012	14.19 ± 0.003
	40B	20B	2	16.39 ± 0.011 (+0.12)	14.37 ± 0.015 (+0.18)
D4	40B	20B	2	**16.10** ± 0.024 (-0.17)	**13.85** ± 0.016 (-0.34)

Table 1. For fixed data selection method and source dataset size, we compare the effects of choosing new tokens or repeating token.. All models are 1.3B OPT models trained on 40B tokens. $T_{selected}$ denotes the number of tokens selected from the source dataset. The top row denotes baseline training. Mean and standard error across 3 seeds are shown. Surprisingly, cleverly choosing tokens to repeat via D4 outperforms randomly selecting new tokens.

et al., 2023) and 22% for OPT-175B (Zhang et al., 2022).

3.4. Analysis of D4

3.4.1. Why does data selection hurt performance on web snapshots?

While we observe consistent *average* perplexity improvements, Section A.3 demonstrates that this perplexity improvement varies greatly across validation sets. More importantly, data selection always impairs performance on web snapshot validation sets such as CC-dedup, CommonCrawl, and C4. To investigate why this occurs, we embed each validation set into the same embedding space as the training set and search for the nearest neighbors to validation points in the training set for our 1.3B baseline model. In the left plot of Figure 5, we show that validation sets derived from the web are substantially closer to training set compared to validation sets derived independently of the web. The right plot of Figure 5 shows that data selection disproportionately affects web-derived validation sets. In the top-right plot, we see that web validation sets reside in regions of the embedding space which are sparsified as a result of data selection (e.g. regions of space close to cluster centroids in the training set), and in the bottom-right plot we see that these points are also the most affected by data selection, since their perplexity after data selection significantly increases. Moreover, the middle-right plot shows that these validation points have the lowest perplexity before pruning indicating that these points are "easy" points, perhaps due to their proximity to the training set.

Given that some of our validation sets are extremely close to the training set, we question whether they are still strong indicators of generalization. In fact, in Figure 6, we find evidence of a slight inverse relationship between perplexity on web snapshots and more robust indicators of LM ability, such as perplexity on instruction-tuned datasets and downstream accuracy. In contrast, we observe that perplexity on Instruct-OPT is positively correlated with downstream accuracy, suggesting that validation perplexity on instruction tuned data is a better measure of model quality. For this reason, we group most of our results in Section 3 into Web Snapshots and Non-web Snapshots.

3.4.2. Importance of re-clustering between SemDeDup and SSL Prototypes

As mentioned in Section 2.4, we hypothesize that sparsifying dense regions of space containing excessive semantic duplicates improves the clustering quality and is, therefore, critical to the performance of D4. To isolate the effect of re-clustering on D4, we run experiments with a version of D4 where we remove the re-clustering step (e.g. we keep the original clustering). As shown in Figure 7, omitting the re-clustering step significantly worsens performance, and we observe in the rightmost plot of Figure 7 that SemDeDup indeed removes extremely dense clusters surrounding centroids (e.g. duplicate-driven clusters). We analyze this in more depth in Section A.5.

4. Related Work

Data selection in non-text domains: Numerous works have successfully used data selection techniques in vision models (Paul et al., 2021; Meding et al., 2021; Chitta et al., 2021; Toneva et al., 2018; Birodkar et al., 2019; Mindermann et al., 2022; Jiang et al., 2019), though these have largely been at sub-ImageNet scale. Some of these works develop pruning metrics that score individual data points (for example, EL2N from Paul et al. (2021)), while some focus on data-efficiency and attempt to find groups of points that allow models to reach baseline performance with less data points, e.g., coresets (Sener & Savarese, 2017; Cazenavette et al., 2022; Zhao et al., 2020; Mirzasoleiman et al., 2020). Sorscher et al. (2022) compares many of the existing individual-score methods at ImageNet scale, finding that their SSL prototypes metrics and the (prohibitively expensive) memorization metric from Feldman & Zhang (2020) generally outperforms other methods. More recently, Abbas et al. (2023) demonstrated very encouraging results on vision-language models (CLIP models) using SemDeDup — a similar method to SSL prototypes but focused on semantic deduplication. Our work combines these approaches and applies them to large-scale LLMs.

Effect of pre-training data on LM performance: Gao et al. (2020) trains variants of GPT-2 (Radford et al., 2019) models from scratch to compare the "Pile" dataset to CommonCrawl-derived corpora. Radford et al. (2019)

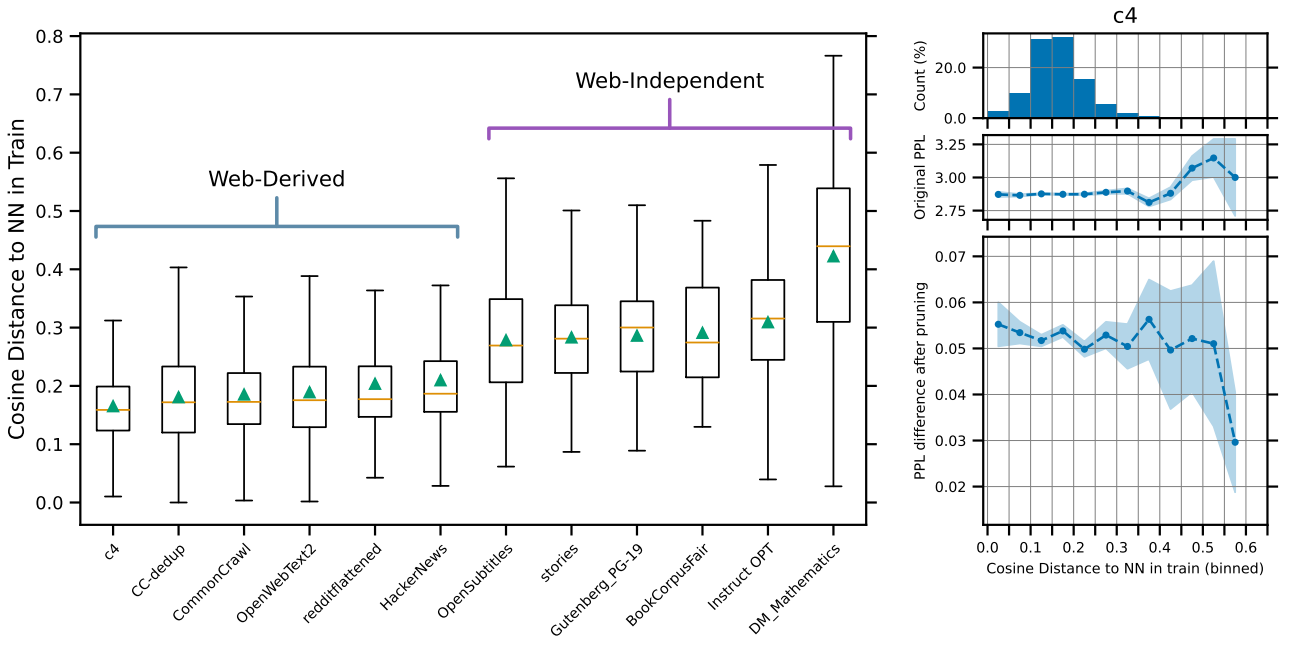

Figure 5. **Left**: Train-test similarity across validation sets. X-axis denotes the name of the validation set (refer to Section 2.4 for more information about each validation set), and y-axis denotes the cosine distance to the nearest neighbor in the training set for the 1.3B OPT 40B baseline. We observe that web-derived validation sets are closer to points in the training set than web-independent validation sets and are disproportionately affected by data selection. **Right**: Analysis of C4 validation set. (Top): Histogram of cosine distance to nearest neighbor in train. For each bin, we show the mean original perplexity (middle) and mean difference in perplexity after data selection (bottom). "Easy" (low original ppl) points close to the training set are generally the points most affected by data selection.

demonstrates the positive impact of the quality filters and data de-duplication methods used to curate MassiveWeb by training 1.4B parameter models from scratch. Hernandez et al. (2022) quantifies the effect of various amounts of artificially created data duplication and provides analysis on interpreting the changes in the behaviour of the models trained on duplicated data. Concurrently to our work, Xie et al. (2023b) propose using importance resampling to align the distribution of web data to high-quality reference corpora such as Wikipedia. Similarly, Gururangan et al. (2020) explores data selection strategies for adapting LMs to a task-specific corpus. Another line of recent work explores how data mixture affects pre-training, with Xie et al. (2023a) demonstrating impressive improvements in downstream accuracy and perplexity across all datasets for 8B parameter models trained on the Pile. Similarly, Longpre et al. (2023) explores the role of text quality, toxicity, age, and domain distribution of training data on LLM performance. Outside of data curation, there has been a recent surge of work exploring the impact of repeating data (Muennighoff et al., 2023; Xue et al., 2023; Biderman et al., 2023), generally concluding that repeating tokens is worse than training on new tokens (which we question in Section 3.2).

5. Summary and Limitations

We introduced D4, a method for data curation on LLMs that improves training efficiency by roughly 20% across multiple model scales, with larger gains at increased model scale. We also demonstrated that, in contrast to comon practice, repeating data via epoching can be beneficial for LLM training, but only if the data subset is intelligently selected. While we have shown encouraging efficiency gains and performance improvements via D4, our work has several limitations and many future directions.

Choice of Embedding Space: The quality of the embedding space and clustering is crucial to the performance of our data selection methods. Due to compute restrictions, we cannot comprehensively investigate the effect of embedding space on data selection. We encourage future work to explore using a different model architecture to generate embeddings: we use OPT models, which are trained on next-word prediction, but we imagine that bidirectional models (e.g., BERT-style models) will give higher quality embeddings. We also primarily work with document embeddings throughout this work and do not explore different document chunking approaches (e.g., selecting data at a paragraph or even sentence level). Most importantly, we

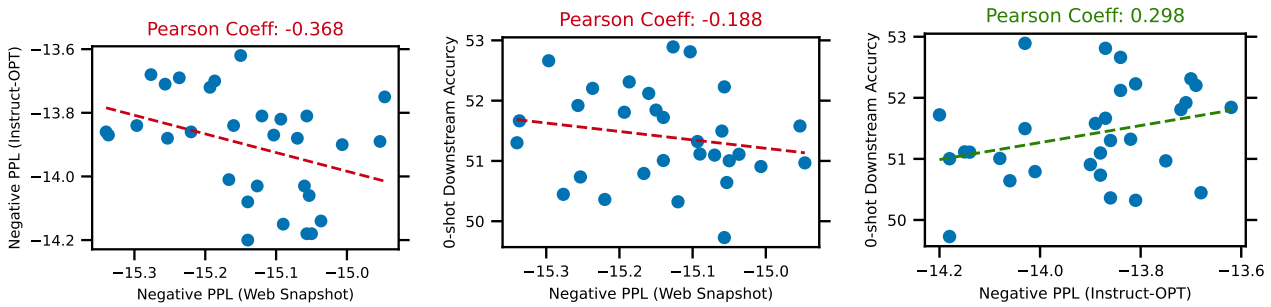

Figure 6. Correlation between (left): negative Instruct-OPT perplexity and negative web snapshot perplexity, (middle): Downstream accuracy and negative web snapshot perplexity, (right): Downstream accuracy and negative Instruct-OPT perplexity. Each point is one training configuration (1.3B OPT model, 40B tokens), with the only change being the data selection method and pretraining seed. Web snapshot perplexity is slightly negatively correlated with stronger indicators of LM ability.

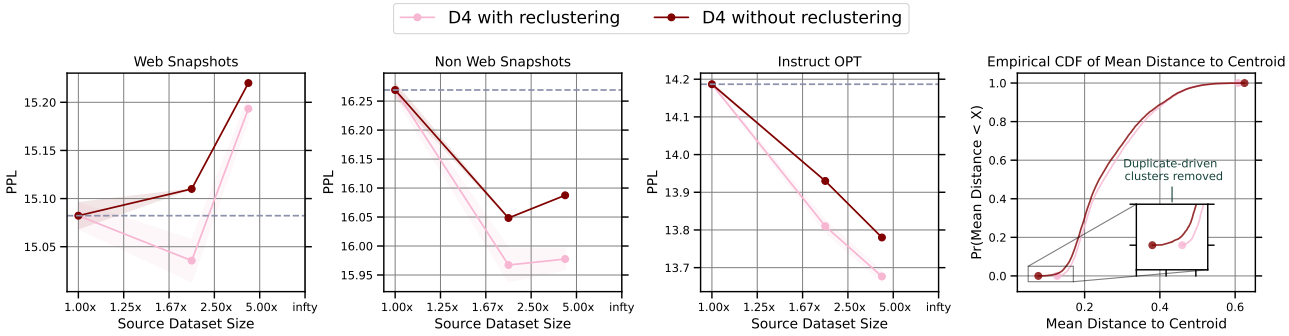

Figure 7. Investigating the necessity of the re-clustering step in D4. We see that re-clustering improves perplexity across Web snapshots (left), Non-web snapshots (middle-left), and Instruct-OPT (middle-right). Right: Empirical CDF of mean distance to centroid, with and without re-clustering. Re-clustering removes duplicate driven clusters (clusters with low mean distance to centroid).

qualitatively observe that our clustering over-emphasized the last tokens in the document. This makes sense given our choice of embedding (we use the last-token embedding for each document), but it has clear downsides, as clustering with a bias towards the end of document will most likely miss clusters of conceptually-related documents.

Mixing different training distributions: While we chose one data distribution to both select data and train on, modern LLM setups usually mix different data sources. Our method is likely complimentary to such pipelines: practitioners may use D4 to diversify and de-duplicate individual data sources and then mix data sources to provide additional diversity in their training dataset. We leave exploring the efficacy of D4 on a mix of training distributions as future work, but expect that this will yield further gains by reducing redundancy across datasets as well as within datasets.

Model scale: Due to compute limitations, the largest models we evaluated were 6.7B parameters trained on 100B tokens. While, to our knowledge, this is the largest to date application of embedding based data curation approaches, further investigation at model scales exceeding 100B would be very interesting, particularly in light of our observation that the efficiency gain grows with model scale.

References

Abbas, A., Tirumala, K., Simig, D., Ganguli, S., and Morcos, A. S. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *ArXiv*, abs/2303.09540, 2023.

Artetxe, M., Bhosale, S., Goyal, N., Mihaylov, T., Ott, M., Shleifer, S., Lin, X. V., Du, J., Iyer, S., Pasunuru, R., et al. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*, 2021.

Bach, S. H., Sanh, V., Yong, Z. X., Webson, A., Raffel, C., Nayak, N. V., Sharma, A., Kim, T., Bari, M. S., Févry, T., Alyafeai, Z., Dey, M., Santilli, A., Sun, Z., Ben-David, S., Xu, C., Chhablani, G., Wang, H., Fries, J. A., Al-shaibani, M. S., Sharma, S., Thakker, U., Almubarak, K., Tang, X., Jiang, M. T.-J., and Rush, A. M. Promptsource: An

integrated development environment and repository for natural language prompts. *ArXiv*, abs/2202.01279, 2022.

Baumgartner, J., Zannettou, S., Keegan, B., Squire, M., and Blackburn, J. The pushshift reddit dataset. In *Proceedings of the international AAAI conference on web and social media*, volume 14, pp. 830–839, 2020.

Biderman, S., Schoelkopf, H., Anthony, Q., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*, 2023.

Birodkar, V., Mobahi, H., and Bengio, S. Semantic redundancies in image-classification datasets: The 10% you don't need. *arXiv preprint arXiv:1901.11409*, 2019.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Broder, A. Z. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pp. 21–29. IEEE, 1997.

Cazenavette, G., Wang, T., Torralba, A., Efros, A. A., and Zhu, J.-Y. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4750–4759, 2022.

Chitta, K., Álvarez, J. M., Haussmann, E., and Farabet, C. Training data subset search with ensemble active learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14741–14752, 2021.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33:2881–2891, 2020.

Gao, L., Biderman, S. R., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The pile: An 800gb dataset of diverse text for language modeling. *ArXiv*, abs/2101.00027, 2020.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hernandez, D., Brown, T. B., Conerly, T., DasSarma, N., Drain, D., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Henighan, T. J., Hume, T., Johnston, S., Mann, B., Olah, C., Olsson, C., Amodei, D., Joseph, N., Kaplan, J., and McCandlish, S. Scaling laws and interpretability of learning from repeated data. *ArXiv*, abs/2205.10487, 2022.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. Training compute-optimal large language models. *ArXiv*, abs/2203.15556, 2022.

Iyer, S., Lin, X., Pasunuru, R., Mihaylov, T., Simig, D., Yu, P., Shuster, K., Wang, T., Liu, Q., Koura, P. S., Li, X., O'Horo, B., Pereyra, G., Wang, J., Dewan, C., Celikyilmaz, A., Zettlemoyer, L., and Stoyanov, V. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. *ArXiv*, abs/2212.12017, 2022.

Jiang, A. H., Wong, D. L.-K., Zhou, G., Andersen, D. G., Dean, J., Ganger, G. R., Joshi, G., Kaminksy, M., Kozuch, M., Lipton, Z. C., et al. Accelerating deep learning by focusing on the biggest losers. *arXiv preprint arXiv:1910.00762*, 2019.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Kaplan, J., McCandlish, S., Henighan, T. J., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lee, K., Ippolito, D., Nystrom, A., Zhang, C., Eck, D., Callison-Burch, C., and Carlini, N. Deduplicating training data makes language models better. In *Annual Meeting of the Association for Computational Linguistics*, 2021.

Levesque, H., Davis, E., and Morgenstern, L. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

Liu, Z., Wang, J., Dao, T., Zhou, T., Yuan, B., Song, Z., Shrivastava, A., Zhang, C., Tian, Y., Re, C., et al. Deja vu: Contextual sparsity for efficient llms at inference time, 2023.

Longpre, S., Yauney, G., Reif, E., Lee, K., Roberts, A., Zoph, B., Zhou, D., Wei, J., Robinson, K., Mimno, D. M., and Ippolito, D. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. *ArXiv*, abs/2305.13169, 2023.

Meding, K., Buschoff, L. M. S., Geirhos, R., and Wichmann, F. A. Trivial or impossible–dichotomous data difficulty masks model differences (on imagenet and beyond). *arXiv preprint arXiv:2110.05922*, 2021.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Mindermann, S., Brauner, J. M., Razzak, M. T., Sharma, M., Kirsch, A., Xu, W., Höltgen, B., Gomez, A. N., Morisot, A., Farquhar, S., et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pp. 15630–15649. PMLR, 2022.

Mirzasoleiman, B., Bilmes, J., and Leskovec, J. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pp. 6950–6960. PMLR, 2020.

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., and Allen, J. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*, 2016.

Muennighoff, N., Rush, A. M., Barak, B., Scao, T. L., Piktus, A., Tazi, N., Pyysalo, S., Wolf, T., and Raffel, C. Scaling data-constrained language models. 2023.

Paul, M., Ganguli, S., and Dziugaite, G. K. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34: 20596–20607, 2021.

Penedo, G., Malartic, Q., Hesslow, D., Cojocaru, R., Cappelli, A., Pannier, B., Almazrouei, E., and Launay, J. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Schaeffer, R., Miranda, B., and Koyejo, S. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*, 2023.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., Catanzaro, B. M.-L., et al. Training multi-billion parameter language models using model parallelism. *arXiv preprint cs.CL/1909.08053*, 2019.

Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

Sorscher, B., Geirhos, R., Shekhar, S., Ganguli, S., and Morcos, A. S. Beyond neural scaling laws: beating power law scaling via data pruning. *ArXiv*, abs/2206.14486, 2022.

Tirumala, K., Markosyan, A., Zettlemoyer, L., and Aghajanyan, A. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35: 38274–38290, 2022.

Toneva, M., Sordoni, A., Combes, R. T. d., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023.

Villalobos, P., Sevilla, J., Heim, L., Besiroglu, T., Hobbhahn, M., and Ho, A. Will we run out of data? an analysis of the limits of scaling datasets in machine learning, 2022.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Arunkumar, A., Ashok, A., Dhanasekaran, A. S., Naik, A., Stap, D., Pathak, E., Karamanolakis, G., Lai, H. G., Purohit, I., Mondal, I., Anderson, J., Kuznia, K., Doshi, K., Patel, M., Pal, K. K., Moradshahi, M., Parmar, M., Purohit, M., Varshney, N., Kaza, P. R., Verma, P., Puri, R. S., Karia, R., Sampat, S. K., Doshi, S., Mishra, S. D., Reddy, S., Patro, S., Dixit, T., Shen, X., Baral, C., Choi, Y., Smith, N. A., Hajishirzi, H., and Khashabi, D. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

Wenzek, G., Lachaux, M.-A., Conneau, A., Chaudhary, V., Guzm'an, F., Joulin, A., and Grave, E. Ccnet: Extracting high quality monolingual datasets from web crawl data. *ArXiv*, abs/1911.00359, 2019.

Xie, S. M., Pham, H., Dong, X., Du, N., Liu, H., Lu, Y., Liang, P., Le, Q. V., Ma, T., and Yu, A. W. Doremi: Optimizing data mixtures speeds up language model pretraining. *ArXiv*, abs/2305.10429, 2023a.

Xie, S. M., Santurkar, S., Ma, T., and Liang, P. Data selection for language models via importance resampling. *ArXiv*, abs/2302.03169, 2023b.

Xue, F., Fu, Y., Zhou, W., Zheng, Z., and You, Y. To repeat or not to repeat: Insights from scaling llm under token-crisis. *arXiv preprint arXiv:2305.13230*, 2023.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer, L. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068, 2022.

Zhao, B., Mopuri, K. R., and Bilen, H. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pp. 19–27, 2015.

A. Appendix

A.1. Experimental Setup Details

A.1.1. Hyperparameters for model training

As mentioned in Section 2.4, we use the same hyperparameters and configurations as the original OPT model architecture from Zhang et al. (2022). We describe these hyperparameters briefly in Table A1. We chose these configurations because they are openly available and have been used as the standard in many previous works (Zhang et al., 2022; Liu et al., 2023; Tirumala et al., 2022; Dettmers et al., 2022; Abbas et al., 2023). All models use GELU activation (Hendrycks & Gimpel, 2016), Adam optimizer (Kingma & Ba, 2014) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay set to 0.1, and we clip gradient norms at 1.0. We use a polynomial learning rate schedule, where learning rate warms up from 0.0 to peak learning rate over the first 375 million tokens, and is then annealed to (0.1 * Peak LR) over the remaining $(T_{target} - 375)$ M tokens. We train all our models in fully sharded data parallel mode (Artetxe et al., 2021) using Megatron-LM Tensor Parallelism (Shoeybi et al., 2019) with fp16 precision. For reproducibility (and perhaps the only difference from the original configuration in Zhang et al. (2022)) is that we do not use dropout.

Table A1. Model architecture details. Most of the parameter configurations are the same as in Table 1 of Zhang et al. (2022). Batch size denotes the total tokens that the model sees during one gradient descent update.

Scale	Num Layers	Num Heads	Embedding Dim	Peak Learning Rate (LR)	Batch Size
8M	4	2	128	1.0e-3	0.5M
125M	12	12	768	6.0e-4	0.5M
1.3B	24	32	2048	2.0e-4	1M
6.7B	32	32	4096	1.2e-4	2M

A.1.2. Dataset Curation Details

In this subsection, we describe how we curate *CC-dedup*, the starting source dataset used throughout the paper. We start with 5 CommonCrawl dumps [3] which range from 2017 to 2020. We then use CC-net (Wenzek et al., 2019), to de-duplicate data at the paragraph level, remove non-English web pages, and filter out low-quality pages. The pipeline we use is identical to the pipeline used in Touvron et al. (2023) (see the section after the subtitle "English CommonCrawl [67%]", within Section 2).

On top of this, we add an additional step of MinHash (Broder, 1997) de-duplication at the document-level. The parameters for MinHash are 20 hashes per signature, 20 buckets, and 1 row per bucket. These parameters are the default parameters in the spark implementation of MinHashLSH, and we did not do a hyperparameter sweep on these parameters due to compute limitations. Previous work has attempted running MinHash with much more aggressive parameters: Lee et al. (2021) and Penedo et al. use 20 buckets, 450 hashes per bucket, and 9000 signatures per hash. We conjecture that more aggressive MinHash would remove more templates, resulting in a higher-quality starting dataset, potentially making the SemDeDup step of D4 less necessary. Abbas et al. (2023) did find that the performance of MinHash from Lee et al. (2021) and SemDeDup are comparable at a fixed data selection ratio of 3.9% on C4, indicating that SemDeDup filters out similar data to aggressive MinHash does. We leave sweeping over these hyperparameters as future work.

We note that since our dataset is curated from CommonCrawl dumps, there is risk that our training set contains offensive or PII content. We note, however, that this risk is no more than that of standard language modeling curation such as Touvron et al. (2023), since we use the same pipeline to filter CommonCrawl dumps.

A.1.3. Parameters for Data Selection

All methods introduced in Section 2.4 involve clustering embeddings using K-Means. Our starting training dataset CC-dedup contains roughly 600 million documents in total. Running K-Means clustering on all 600 million 768-sized vectors would take a considerable amount of compute. Instead, we follow previous work (Sorscher et al., 2022; Abbas et al., 2023) and randomly sample roughly 100M documents with which to calculate centroids. We normalize the embeddings for these 100M documents to have L2-norm of 1.0, and then use faiss (Johnson et al., 2019) with the following parameters:

[3] https://commoncrawl.org/the-data/get-started/

```
faiss.Kmeans(
    768 # 125M OPT model embedding size,
    11000 # 11K clusters,
    niter=20 # 20 iterations,
    verbose=True,
    seed=0,
    gpu=False,
    spherical=True,
    min_points_per_centroid=1,
    max_points_per_centroid=100000000
)
```

We choose 11000 clusters following previous work (Abbas et al., 2023) and we note that this choice sticks to the heuristic that the number of clusters should roughly be the square root of the number of total points being clustered. We also note that in initial experiments for data selection at the 125M OPT model scale, we did not find a significant effect of number of clusters on the performance of our data selection methods (see Figure A1) this finding agrees with Abbas et al. (2023) who notice significant overlap between datasets selected by SemDeDup with different number of clusters (see Figure A2 in Abbas et al. (2023)).

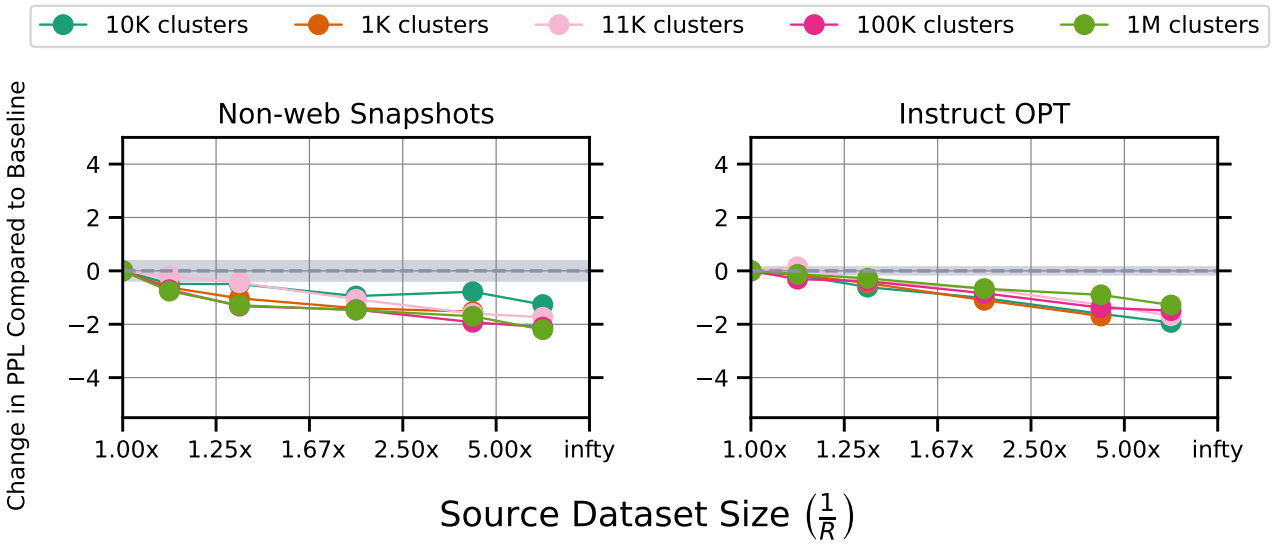

Figure A1. Effect of number of clusters in K-Means on data selection performance. All models are 125M OPT models, where the training set (and starting source dataset) is C4 and we select data with SSL prototypes. The y-axis is the change in perplexity compared to baseline training, meaning that baseline training is at 0.0, and going *down* on the graphs indicates *better* performance. The x-axis is the source dataset size. We show results for average perplexity on Non-web snapshot validation sets (left) and Instruct-OPT (right). We notice that there is not a significant difference when changing number of clusters (e.g. if we drew error bars around each line, they would all be overlapping), but 11K clusters is generally among the top-3 best performing methods.

We deliberately set min points per centroids low and max points per centroid high so that faiss does not attempt to manually balance the clusters while doing K-Means. Sorscher et al. (2022) found that explicitly class-balancing is important: they introduce the "class balance score" (see Section H of Sorscher et al. (2022)) which is the expectation of the quantity $\frac{\text{size of majority class}}{\text{size of minority class}}$ over all pairs of classes. They then set a hard limit for the class balance score of 0.5, meaning that "every class has at least 50% of the images that it would have when pruning all classes equally" (Sorscher et al., 2022). We consider the unsupervised-learning analog of the class-balance score, which we refer to as the "cluster balance" score. The cluster balance score is the expectation of the quantity $\frac{\text{size of bigger cluster}}{\text{size of smaller cluster}}$ over all pairs of clusters. Across all of our data selection methods (and choices for R) we find that this value is generally equal to or bigger than 0.5 without any explicit intervention. For this reason, we do not explicitly cluster balance, although we note that changing how many points are sampled from each cluster (based on properties of the cluster) is very interesting future work.

D4 parameters: The choice of parameters R_{proto} and R_{dedup} while using D4 will have impact on the performance of D4. Given limited compute, we are not able to sweep over these hyperparameters. Instead, we strategically choose these parameters: we first look at the highest value of R in SemDeDup that results in perplexity improvement across validation sets. We choose the "highest value" because the purpose of SemDeDup is to remove duplicate-driven clusters and low R with SemDeDup generally removes more than just templates/semantic duplicates. As seen in Section A.3, this generally occured with $R_{dedup} = 0.75$. Thus, we chose $R_{dedup} = 0.75$ and varied R_{proto} to obtain different data selection ratios for D4.

A.1.4. WHICH VALIDATION SETS GO INTO THE AVERAGES?

For clarity, we explicitly state the validation sets which we consider "Web Snapshots", "Non Web Snapshots", and "Instruct OPT" when reporting averages:

Web Snapshots: perplexity on validation set of C4, CC-dedup, CommonCrawl (from the Pile)

Non-web Snapshots: perplexity other validation sets from the Pile, comprising of OpenWebText2, HackerNews, Wikipedia (en), BookCorpusFair, DM Mathematics, Gutenberg PG-19, OpenSubtitles, and USPTO. Also included in this average is "redditflattened" (validation set from Pusshift.io Reddit (Baumgartner et al., 2020)), "stories", "prompts_with_answers" (which is described below) and "prompts" (which is the same as "prompts_with_answers" but where each sample is just the instruction-tuning prompt without the answer).

Instruct-OPT: perplexity on instruction-tuning data from OPT-IML (Iyer et al., 2022), where each sample contains both the instruction-tuning prompt and the answer (in Figure A4 this is referred to as "prompts_with_answers."

A.2. Efficiency gains across model scales and training

In this section, we investigate the relationship between model scale, and performance gain obtained by selecting data via D4. Specifically, we train three groups of models: 125M OPT models trained on $T_{target} = 3$B tokens, 1.3B OPT models trained on $T_{target} = 40$B tokens, and 6.7B OPT models trained on $T_{target} = 100$B tokens. We notice in Figure A2 that D4 results in efficiency gains across the board in terms of perplexity. Surprisingly, these efficiency gains seem to increase with scale, indicating that at bigger model scales, D4 might lead to even more efficiency gains. We also see efficiency gains in 0-shot downstream accuracy for 1.3B and 6.7B model scales on the order of 30% for both 1.3B and 6.7B models, but we note that evaluation downstream performance on intermediate checkpoints is not completely fair due to unfinished learning rate schedule. Nonetheless, we see that downstream accuracy efficiency gains are not decreasing with scale.

A.3. Individual Breakdowns of Downstream Accuracy and PPL

In Section 3, we see that D4, SSL prototypes, and SemDeDup achieves significant gains on perplexity (averaged across different validation sets) and downstream accuracy (averaged across different NLP tasks) compared to baseline training. Further, we generally see that D4 outperforms SSL prototypes and SemDeDup. In this section, we provide a more fine-grained analysis of these claims across individual tasks.

For perplexity, we notice in Figure A4 that the claims in Section 3 generally hold across validation sets: for web snapshots validation sets such C4, CC-dedup, and CommonCrawl, we see performance worsens with data selection compared to baseline training, and that D4 generally has the slowest rate of performance degradation. We note that, across all non web-snapshot validation sets, there is no clear winner among data selection methods. We emphasize however that *we observe consistent improvement over baseline training on most validation sets* we use — for example in Figure A4 we observe that, when selecting tokens from a 1.25x source dataset, all data selection methods improve over baseline across all validation sets except C4 and CC-dedup (however, as we explain in Section 3.4, this decrease in performance on C4 and CC-dedup is expected).

For downstream accuracy, we chose to match the exact downstream evaluation done in Zhang et al. (2022) since we use OPT architecture and hyperparameters. Similar to Zhang et al. (2022), we notice considerable variability across the 16 NLP tasks in Figure A3, motivating us to look at the mean downstream accuracy across tasks.

A.4. SSL prototypes and SemDeDup overlap

Figure A5 shows the overlap between datasets selected by SemDeDup and SSL Prototypes. While the two methods do not arrive at the same set of data points, there is a significant overlap between the datasets curated by the two methods. We hypothesize that this is because both SSL prototypes and SemDeDup prune away dense regions of space surrounding cluster centroids: by definition, SemDeDup sparsifies dense regions of space within a cluster; similarly, by definition, SSL prototypes will prune away datapoints close to the cluster centroids. Since K-means clustering places centroids in dense regions of space (see Figure A6 where we observe that the distribution of cosine distances to cluster centroid is skewed right), we know that the regions of space surroundings centroids will be dense, and expect SSL prototypes and SemDedup to have significant overlap. Qualitatively, we inspect a few examples of points close to cluster centroids in Figure A2, Figure A3, Figure A4, and see that examples close to cluster centroids can be semantically redundant (e.g. templates). Therefore, it makes sense that any reasonable data selection strategy would prioritize sparsifying these dense regions of space surrounding cluster centroids. As mentioned in Section 2.4, sparsifying these dense regions of space containing excessive semantic duplicates is the original motiviation behind D4. As shown in Figure 7, omitting the re-clustering step significantly worsens performance, and we observe in the rightmost plot of Figure 7 that SemDeDup indeed removes duplicate-driven clusters.

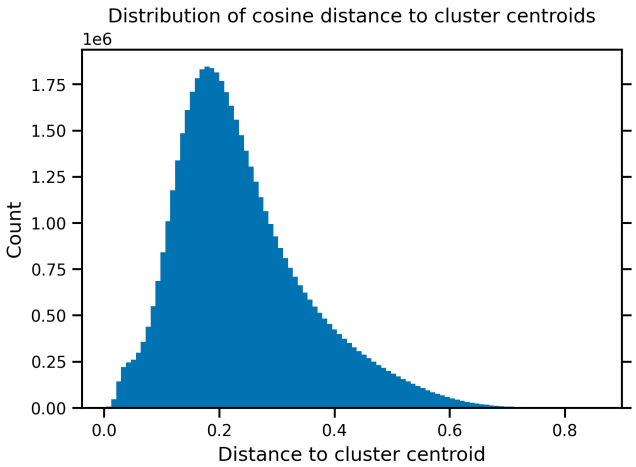

Figure A6. Distribution of cosine distance to cluster centroids for 50M randomly selected documents from the training set of CC-dedup. We notice that the distribution is skewed right, implying that datapoints are generally close to centroids.

Table A2. Nearest Neighbors to Cluster Centroid 682

Cosine Distance to Centroid	Raw Text
0.03581655	The USGS (U.S. Geological Survey) publishes a set of the most commonly used topographic maps of the U.S. called US ..... may have differences in elevation and topography, the historic weather at the two separate locations may be different as well.
0.03584063	The USGS (U.S. Geological Survey) publishes a set of the most commonly used topographic maps of the U.S. called US ..... may have differences in elevation and topography, the historic weather at the two separate locations may be different as well.
0.036803484	The USGS (U.S. Geological Survey) publishes a set of the most commonly used topographic maps of the U.S. called US ..... may have differences in elevation and topography, the historic weather at the two separate locations may be different as well.
0.037270606	Search Near Clinton County, OH: Trails National and State Parks City Parks Lakes Lookouts Marinas Historical Sites The USGS (U.S. Geological ..... may have differences in elevation and topography, the historic weather at the two separate locations may be different as well.

Table A3. Nearest Neighbors to Cluster Centroid 975

Cosine Distance to Centroid	Raw Text
0.011662006	The American Way, Inc. The American Way, Inc. is a suspended Californian business entity incorporated 19th August 1949. is listed as ..... for bulk data downloadsI want to request the removal of a page on your websiteI want to contact California Explore
0.012483656	John St-Amour, Inc. John St-Amour, Inc. is a suspended Californian business entity incorporated 5th October 1962. is listed as the agent ..... for bulk data downloadsI want to request the removal of a page on your websiteI want to contact California Explore
0.012564898	Joseph E. Barbour, Inc. Joseph E. Barbour, Inc. is a suspended Californian business entity incorporated 27th January 1959. is listed as ..... for bulk data downloadsI want to request the removal of a page on your websiteI want to contact California Explore
0.012756169	The Jolly Boys, Inc. The Jolly Boys, Inc. is a suspended Californian business entity incorporated 4th March 1955. is listed as ..... for bulk data downloadsI want to request the removal of a page on your websiteI want to contact California Explore

Table A4. Nearest Neighbors to Cluster Centroid 10715

Cosine Distance to Centroid	Raw Text
0.035506427	Search hundreds of travel sites at once for hotel deals at Hotel Olympic Kornarou Square 44, Heraklion, Greece 34 m Bembo Fountain 262 ..... hundreds of travel sites to help you find and book the hotel deal at Hotel Olympic that suits you best.
0.036230028	Search hundreds of travel sites at once for hotel deals at Hotel Estrella del Norte Juan Hormaechea, s/n, 39195 Isla, Cantabria, ... travel sites to help you find and book the hotel deal at Hotel Estrella del Norte that suits you best.
0.036280274	Search hundreds of travel sites at once for hotel deals at H10 Costa Adeje Palace Provided by H10 Costa Adeje Palace Provided ..... travel sites to help you find and book the hotel deal at H10 Costa Adeje Palace that suits you best.
0.036827266	Search hundreds of travel sites at once for hotel deals at Hotel Miguel Angel by BlueBay Calle Miguel Angel 29-31, 28010 ..... sites to help you find and book the hotel deal at Hotel Miguel Angel by BlueBay that suits you best.

A.5. Investigating Duplicate-Driven Clusters

In this subsection, we present a few examples of duplicate-driven clusters, which are clusters that are very dense and near centroids. We find that these clusters tend to be filled with semantic duplicates and/or duplicated text. We generally can find such extreme duplicate-driven clusters by looking at clusters whose standard deviation of cosine distance to cluster centroid is less than 0.03. This is essentially looking at clusters in the lower tail of the empirical CDF in Figure 7 (brown line). We present a few examples of such clusters below:

Table A5. Random Examples from Cluster 695

Cosine Distance to Cluster Centroid	Raw Text
0.044178426	Eastern Florida State College nutritional sciences Learn about Eastern Florida State College nutritional sciences, and registering for electives. Which college degrees ..... System (IPEDS). If any stats on Hagerstown Community College career planning are incorrect, please contact us with the right data.
0.056984067	Albany State University introduction to business Find info concerning Albany State University introduction to business, and registering for elective discussion sections ..... If any stats on Warren County Community College plant science major are incorrect, please contact us with the right data.
0.0534693	Baldwin Wallace University cost per unit Learn about Baldwin Wallace University cost per unit, submitting required application forms, and follow-up scheduling. ..... (IPEDS). If any stats on San Jose State nursing degree programs are incorrect, please contact us with the right data.
0.06892538	Niagara University managerial accounting Information about Niagara University managerial accounting, and registering for elective lectures. Which college degrees give you the ..... System (IPEDS). If any stats on Midwestern University pharmacy tech program are incorrect, please contact us with the right data.
0.07246786	Fanshawe College app download Learn about Fanshawe College app download, and registering for elective discussion sections and seminars. Which college degrees ..... Data System (IPEDS). If any stats on Stratford University cell biology are incorrect, please contact us with the right data.
0.07147932	Standish Maine Licensed Vocational Nurse LVN Jobs Find out about Standish, ME licensed vocational nurse LVN jobs options. It's a smart ..... (IPEDS). If any stats on William Jewell College medical insurance coding are incorrect, please contact us with the right data.

Table A6. Random Examples from Cluster 8342

Cosine Distance to Cluster Centroid	Raw Text
0.027729392	Seenti - Bundi Seenti Population - Bundi, Rajasthan Seenti is a medium size village located in Bundi Tehsil of Bundi district, Rajasthan ..... 6 months. Of 186 workers engaged in Main Work, 63 were cultivators (owner or co-owner) while 0 were Agricultural labourer.
0.036407113	Kodunaickenpatty pudur - Salem Kodunaickenpatty pudur Population - Salem, Tamil Nadu Kodunaickenpatty pudur is a large village located in Omalur Taluka of ..... 6 months. Of 3523 workers engaged in Main Work, 1500 were cultivators (owner or co-owner) while 1533 were Agricultural labourer.
0.017463684	Chhotepur - Gurdaspur Chhotepur Population - Gurdaspur, Punjab Chhotepur is a medium size village located in Gurdaspur Tehsil of Gurdaspur district, Punjab ..... 6 months. Of 677 workers engaged in Main Work, 123 were cultivators (owner or co-owner) while 142 were Agricultural labourer.
0.02616191	Maksudanpur - Azamgarh Maksudanpur Population - Azamgarh, Uttar Pradesh Maksudanpur is a small village located in Sagri Tehsil of Azamgarh district, Uttar ..... 6 months. Of 22 workers engaged in Main Work, 14 were cultivators (owner or co-owner) while 0 were Agricultural labourer.
0.028420448	Karambavane - Ratnagiri Karambavane Population - Ratnagiri, Maharashtra Karambavane is a medium size village located in Chiplun Taluka of Ratnagiri district, Maharashtra ..... 6 months. Of 444 workers engaged in Main Work, 116 were cultivators (owner or co-owner) while 214 were Agricultural labourer.
0.037917078	Barda - Purba Medinipur Barda Population - Purba Medinipur, West Bengal Barda is a large village located in Egra - I Block ..... 6 months. Of 1182 workers engaged in Main Work, 278 were cultivators (owner or co-owner) while 252 were Agricultural labourer.

A.6. Investigating Train-Validation overlap

As briefly described in Section 3.4, we observe that many of our validation sets are close (in cosine distance) to our training sets, and the impact of data selection is varies across individual validation sets. Individual validation sets live in different regions of the embedding space, and as such they are affected differently by data selection. For example, one could imagine that web-snapshot validation sets such as C4 is close to CC-dedup in the embedding space, while esoteric validation sets (such as Gutenberg PG 19 or DM Mathematics) might be far. To quantify this, we first find the nearest neighbors in the training set to each validation point in all of our validation sets. We then qualitatively check (see Table A7 and Table A8 for examples) that nearest-neighbors in the training set truly convey information about validation points. we observe significant overlap between training points and validation points. We then quanitatively analyze how close each validation set is to the training set: in Figure A8, we show the breakdown of this distribution for each validation set. We see a general trend, that web-snapshots validation sets are closest to the training set as they are skewed to the right, while more esoteric validation sets (Gutenberg, or Wikipedia (en)) are more centered or even slightly left-skewed.

Motivated by this, we compare validation sets side-by-side (in terms of distance to training set) in Figure 5, and we see a similar trend. To further understand why different validation sets are affected differently by data selection, we loop through each data point in the validation set and record:

- distance to the training set e.g. how close is the validation point to the training set
- perplexity difference before and after data selection with D4 e.g. how much was this validation point affected by data

selection

- original perplexity e.g. how easy was this data point originally

In Figure A7, we observe an interesting trend: for web-snapshot validation sets such as C4, the validation points closest to the training set are both (1) the easiest (lowest perplexity) points before data selection and (2) the points most affected by data selection. This seems to indicate that these validation points are ”easy” due to their proximity to training points, and when these training points are removed from the training set due to data selection, the close-by validation points become difficult for the model. We do not see this trend on non-web snapshot validation sets such as DM Mathematics and Open Subtitles; in fact, we see an opposite trend where points furthest from the training set are generally most affected by data selection.

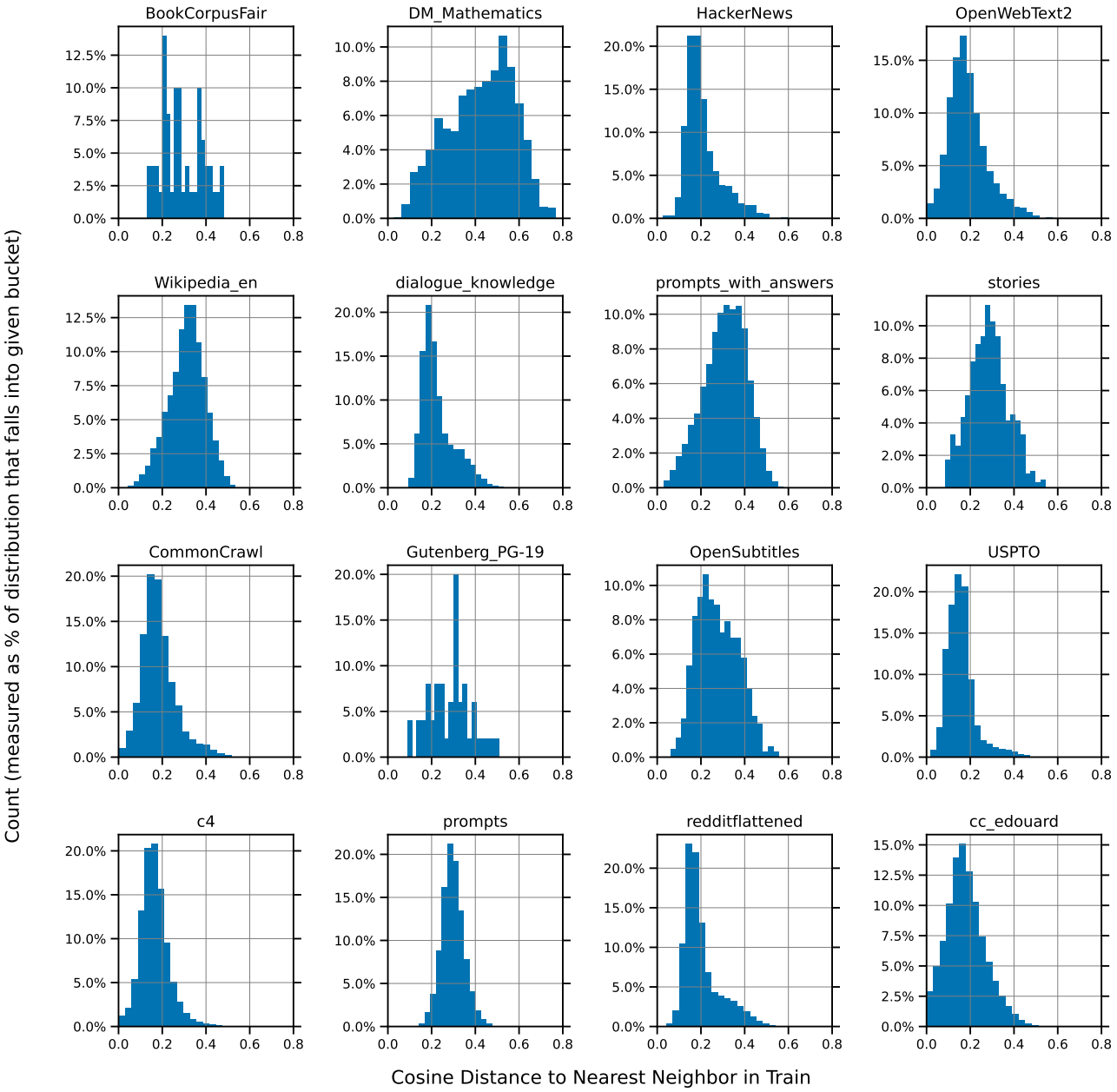

Figure A8. Distribution of cosine distance to nearest neighbor in the training set, for each individual validation set.

Table A7. Nearest Neighbors to random validation point in C4

Cosine Distance	Raw Text
0.0(original validation text)	Offers two child care opportunities to Charles County citizens— the Port Tobacco Onsite Child Care Program and the Before and After School Child Care Program (BASCC). Supports parents through home visits to first time parents and by helping them search for child care, find resources for a child with social, emotional . . . . . Special needs kids. Free to look, a fee to contact the providers. Hotline is staffed by highly-trained and friendly Child Care Consumer Education Specialists who offer both parents and providers invaluable information about child care, and referrals to local Child Care Resource and Referral agencies where they can receive individualized assistance.
0.12867724895477295	Child Care Options is a program of Options Community Services , a non-profit registered charity dedicated to making a difference in the South Fraser Region. Options is committed to empowering individuals, supporting families and promoting community health. Funding for Child Care Options is provided through British Columbia's Ministry of Children . . . . . Rock. Child Care Options links families and child care providers in the communities of Delta, Surrey and White Rock by offering free consultation, support and child care referral services and subsidy support to parents seeking child care. Child care providers are supported through information, outreach, resource library, networking, and learning opportunities.
0.15080827474594116	Below are links to child development resources, both from within the department and from external sources. Child Development Division Publications Publications that can help you will help you follow your child's development (from birth to age five) so you can identify and address any issues early on. Resources to help you understand children's . . . . . families to local resources and services. Specialists are available from 9 AM to 6 PM Monday – Friday. Services are confidential. Caregivers can also visit http://www.helpmegrowvt.org/families.html to learn more about child development, discover developmental tips, and watch videos demonstrating children's developmental milestones (click a button to choose your child's age).
0.15738284587860107	National Domestic Violence Hotlines Programs that provide immediate assistance for women and men who have experienced domestic abuse which may include steps to ensure the person's safety; short-term emotional support; assistance with shelter; legal information and advocacy; referrals for medical treatment; ongoing counseling and/or group support; and other related services. Hotline . . . . . RP-1500.1400-200) www.thehotline.org/ Toll Free Phone: 800-799-SAFE URL: https://www.thehotline.org/ Eligibility: Anyone affected by relationship abuse. Services Provided: Available 24/7/365 via phone, TTY, and chat. Provides lifesaving tools and immediate support to enable victims to find safety and live lives free of abuse. Highly trained, experienced advocates offer support, crisis intervention, education, safety planning, and referral services.

Table A8. Nearest Neighbors to random validation point in USPTO

Cosine Distance	Raw Text
0.0(original validation text)	SONET (Synchronous Optical NETwork) is a North American transmission standard for optical communication systems. SDH (Synchronous Digital Hierarchy), a European transmission standard, is a minor variant of SONET. SONET defines a hierarchy of electrical signals referred to as Synchronous Transport Signals (STS). The STS hierarchy is built upon a basic signal . . . . . the corresponding row and column numbers may include up to 18 comparison operations, which are onerous to implement, for example, in terms of the required logic circuitry. This problem is exacerbated at the upper levels of the STS hierarchy, where processing of multiple pointer values per data frame is performed.
0.1998944878578186	US20080109728A1 - Methods and Systems for Effecting Video Transitions Represented By Bitmaps - Google Patents Methods and Systems for Effecting Video Transitions Represented By Bitmaps Download PDF David Maymudes Multi-media project editing methods and systems are described. In one embodiment, a project editing system comprises a multi-media editing application that is configured to . . . . . synchronization models for multimedia data US20120206653A1 (en) 2012-08-16 Efficient Media Processing US6658477B1 (en) 2003-12-02 Improving the control of streaming data through multiple processing modules US6212574B1 (en) 2001-04-03 User mode proxy of kernel mode operations in a computer operating system US7752548B2 (en) 2010-07-06 Features such as titles, transitions, and/or effects which vary according to positions
0.21122217178344727	Both the Ethernet II and IEEE 802.3 standards define the minimum frame size as 64 bytes and the maximum as 1518 bytes. This includes all bytes from the Destination MAC Address field through the Frame Check Sequence (FCS) field. The Preamble and Start Frame Delimiter fields are not included when . . . . . frame. Dropped frames are likely to be the result of collisions or other unwanted signals and are therefore considered invalid. At the data link layer the frame structure is nearly identical. At the physical layer different versions of Ethernet vary in their method for detecting and placing data on the media.
0.2133803367614746	A byte is a group of bits, usually eight. As memory capacities increase, the capacity of chip cards is often quoted in bytes rather than in bits as in the past.

A.7. Further investigation of repeating tokens

In this section, we investigate whether the findings from Section 3.2 hold across model scale, data selection ratio (e.g. number of epochs), and data selection method.

Across data selection methods: We first take the same configuration as Section 3.2, where we have a starting source dataset of 40B tokens, use each of our data selection methods with $R = 0.25$ to select a subset of documents, and repeat over these documents until we reach the target token budget of 40B tokens. Note that this is at the 1.3B model scale. In Figure A9 we see that repeating data selected by both SemDeDup and SSL prototypes also outperforms randomly selecting new data. However, we quickly notice that for *fixed* data selection strategy (e.g. *fixed* column in Figure A9), repeating tokens either outperforms or matched selecting new tokens. In other words: cleverly repeating tokens can outperform randomly selecting new tokens, but if we fix the data selection strategy (random, SemDeDup, SSL prototypes, or D4) then it is usually preferable to select new tokens.

Across model scale and data selection ratio: We fix our data selection strategy as D4 as done in Section 3.2, but attempt repeating tokens across 3 model scales (125M, 1.3B, and 6.7B), and across data selection ratios ($R = 0.5$ and $R = 0.25$). We see in Figure A11 that repeating data with D4 outperforms randomly selecting new tokens across all model scales and choice of R.

We note that for fixed R, different data selection methods will choose subsets of the source dataset that contain different amounts of tokens. This means that different data selection methods will epoch a different number of times. For example, for a 1.3B OPT model 40B token budget training run, if randomly repeating data with $R = 0.25$ chooses a subset with 10B tokens and D4 with $R = 0.25$ chooses a subset with 15B tokens, then the random run will epoch 4 times while the D4 run will epoch 2.67 times. To show this more clearly, we plot 1.3B and 6.7B repeated data runs with the x-axis changed to number of epochs in Figure A10. We see that up to roughly 2 epochs of data chosen with D4 significantly outperforms randomly selected new data; however, close to 5 epochs leads to worse performance.

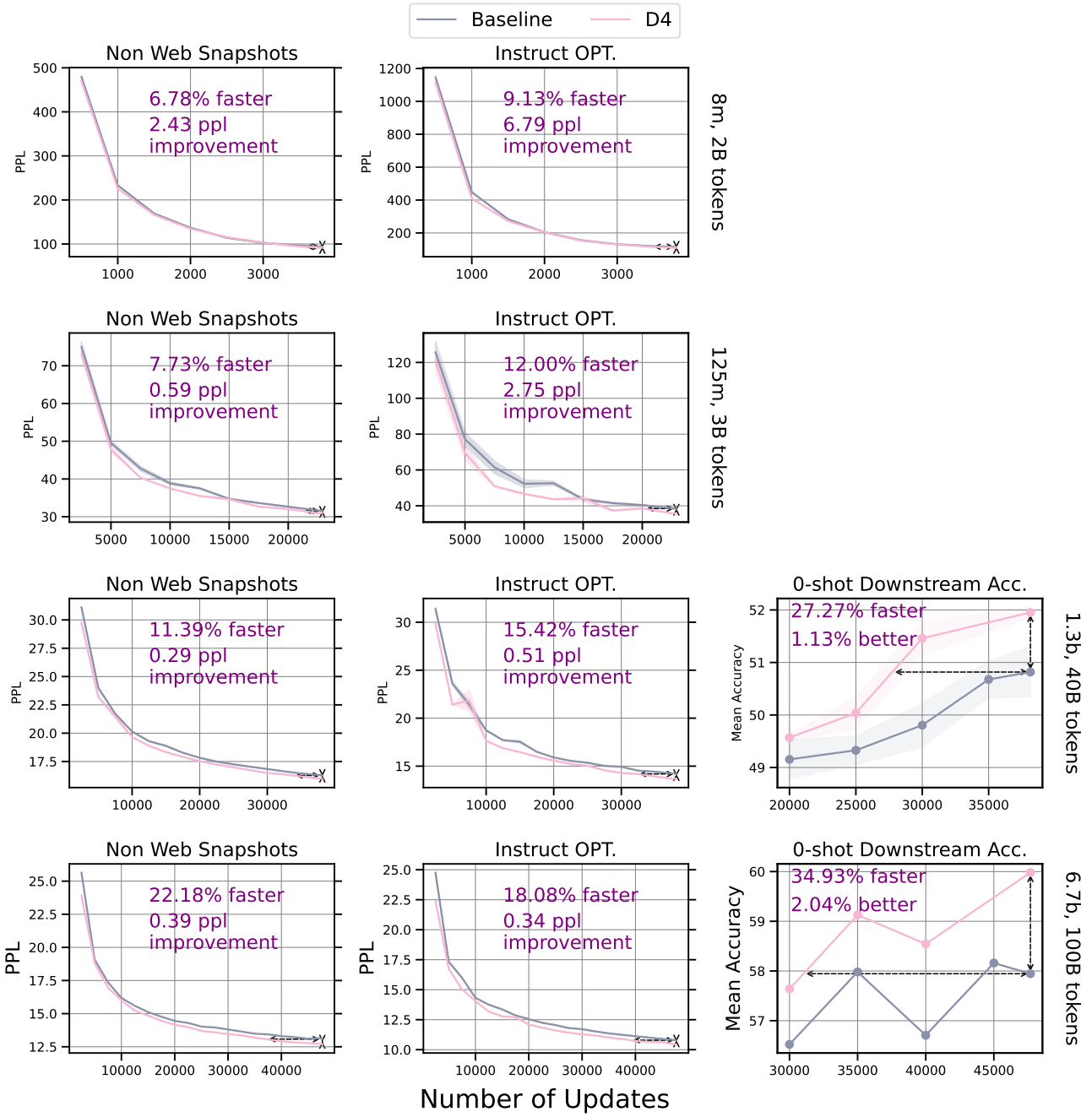

Figure A2. Training trajectory of OPT models trained on raw data (gray line) and data selected via D4 (pink line). Across model scales (1st row: 8M OPT models trained on 2B tokens, 2nd row: 125M OPT models trained on 3B tokens, 3rd row: 1.3B OPT models trained on 40B tokens, 4th row: 6.7B OPT models trained on 100B tokens), we see significant efficiency gains in both perplexity (left two columns) and 0-shot downstream accuracy on 16 NLP tasks (right column). Importantly, we see that increasing model scale does not decrease efficiency gains. All plots show mean and standard error across three seeds, except for the last row. We do not evaluate downstream accuracy for models smaller than 1.3B because they are likely too close to random performance to indicate whether a particular data selection method is better.

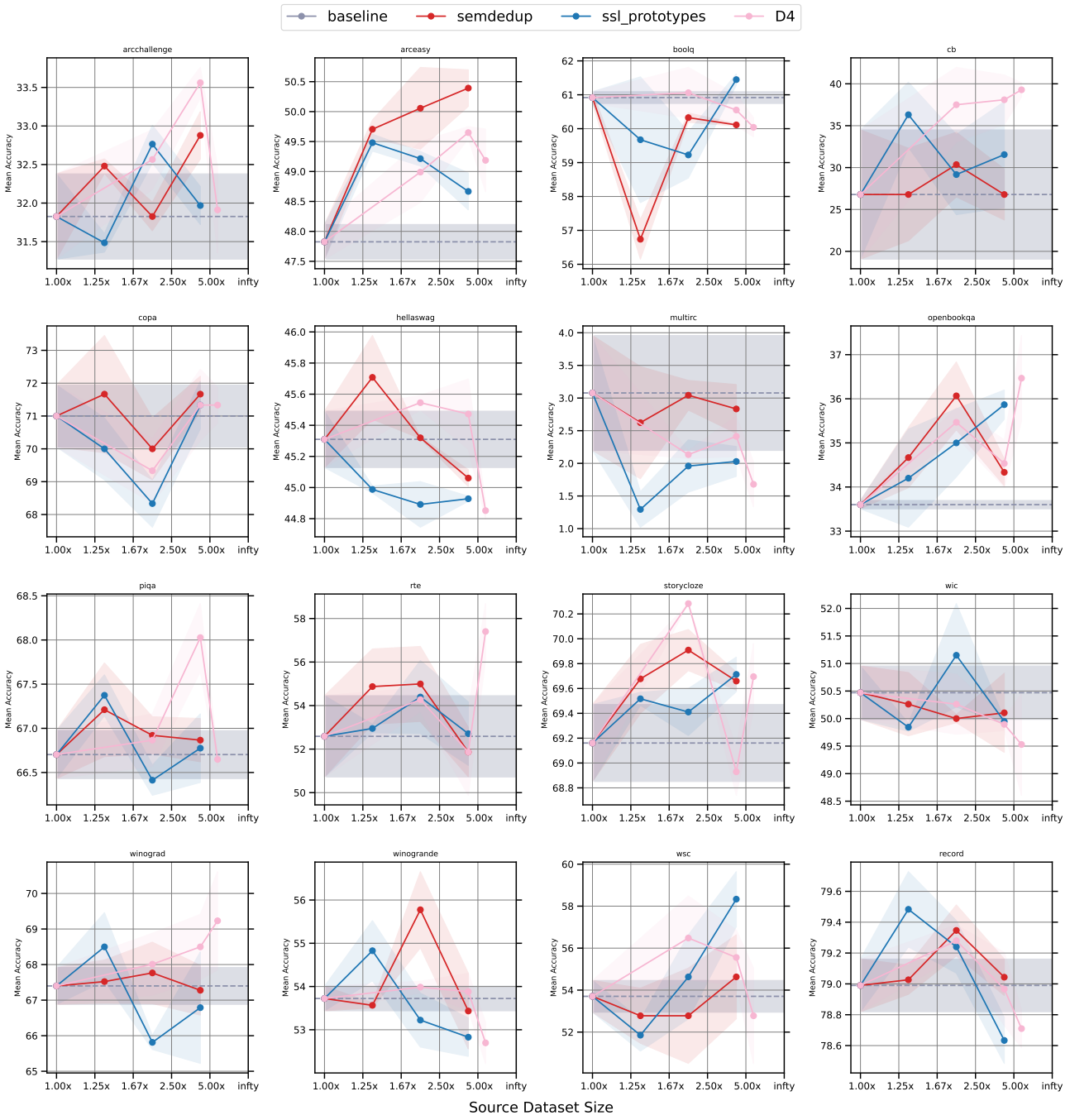

Figure A3. Per-task breakdown of 0-shot downstream accuracy comparison across data selection methods, for 1.3B, 40B OPT model. For a description of the 16 NLP tasks shown above, see Section 2.4. We note that there is considerable variability across individual downstream tasks.

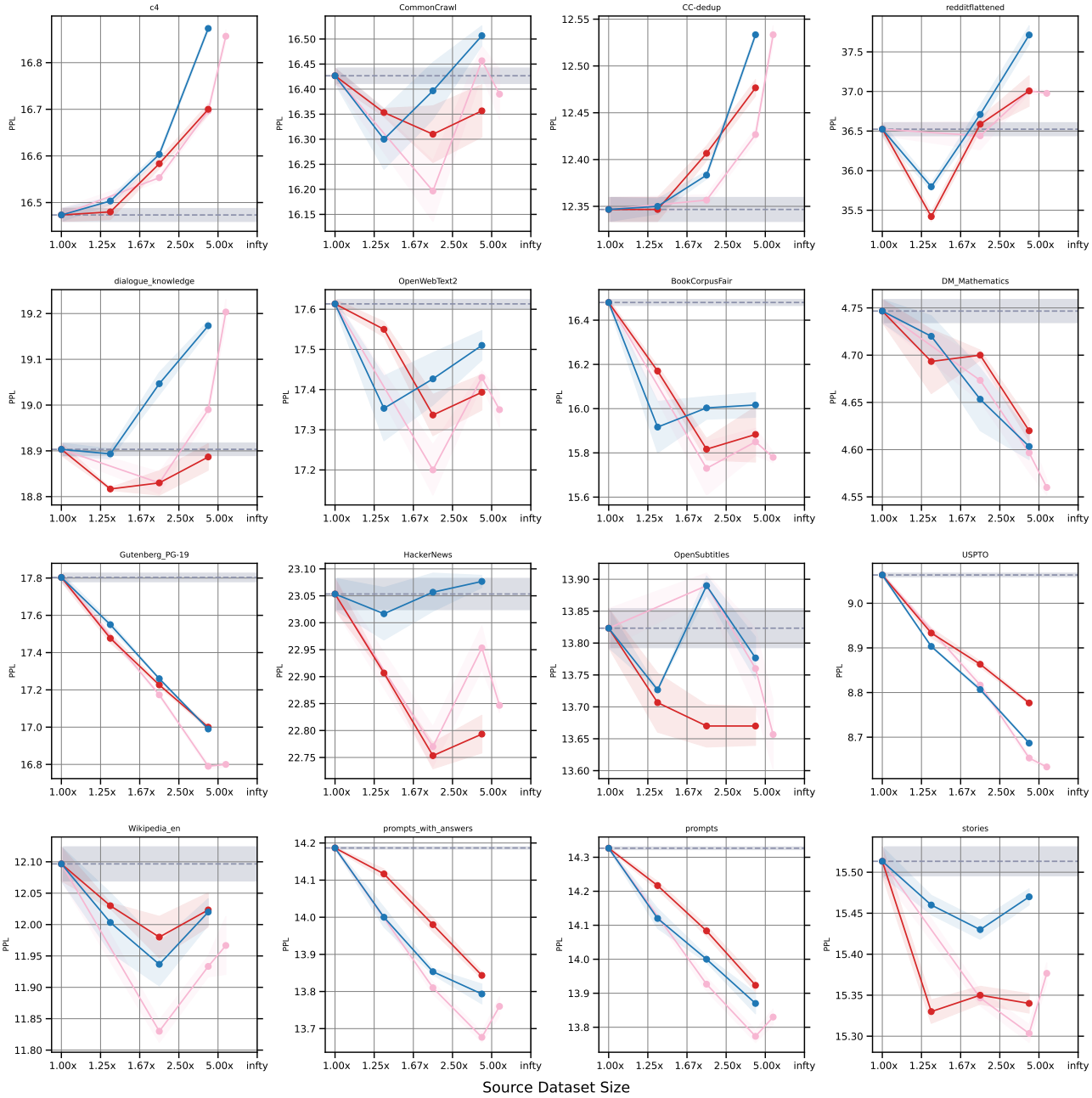

Figure A4. Perplexity as a function of source dataset size for 1.3B OPT model 40B token training runs, across data selection runs. Each plot above represents perplexity on an individual validation set (see Section 2.4 for more information). Mean and standard error across 3 seeds is shown (standard error is denoted by shaded regions).

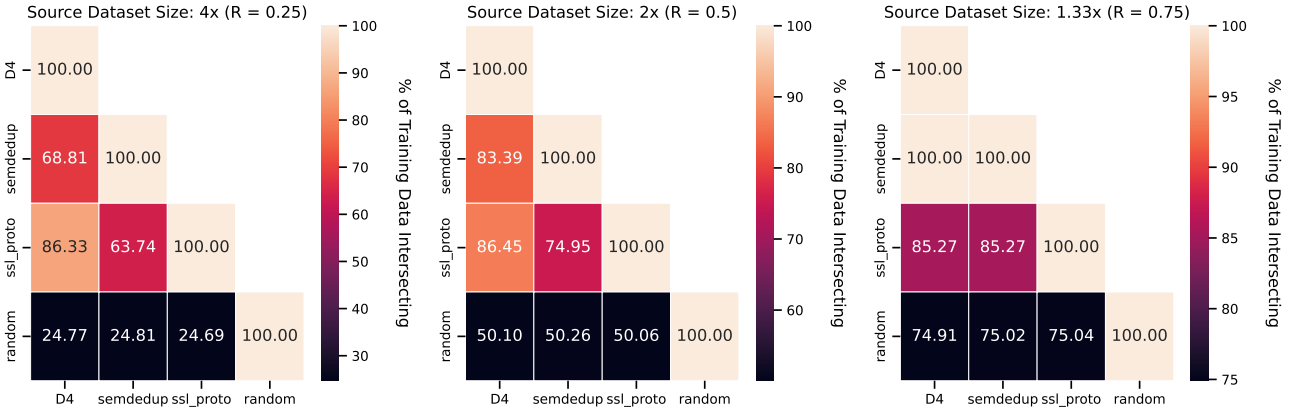

Figure A5. Similarity between data selection methods. Each square represents the percentage of training data that is intersecting, when selecting data via two different strategies. The x and y axis enumerate different data selection strategies.

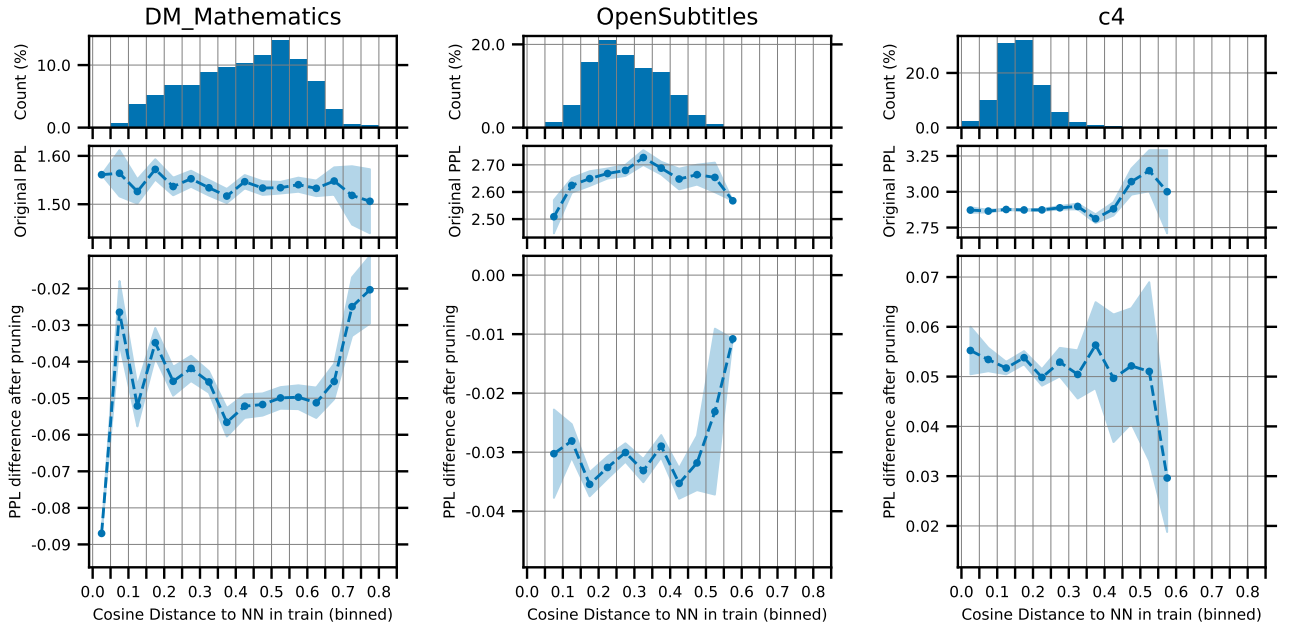

Figure A7. (Top): Histogram of cosine distance to nearest neighbor in train. Within each bin, we show the mean original perplexity (middle) and mean difference in perplexity after data selection (bottom), for DM_Mathematics (left), OpenSubtitles(middle), and C4 (right). We note that points in the C4 validation set closest to the training set are both "easy" (perhaps because of proximity to training points) and are affected the most by data selection. We do not see this trend for non-web snapshot validation sets such as DM_Mathematics and OpenSubtitles.

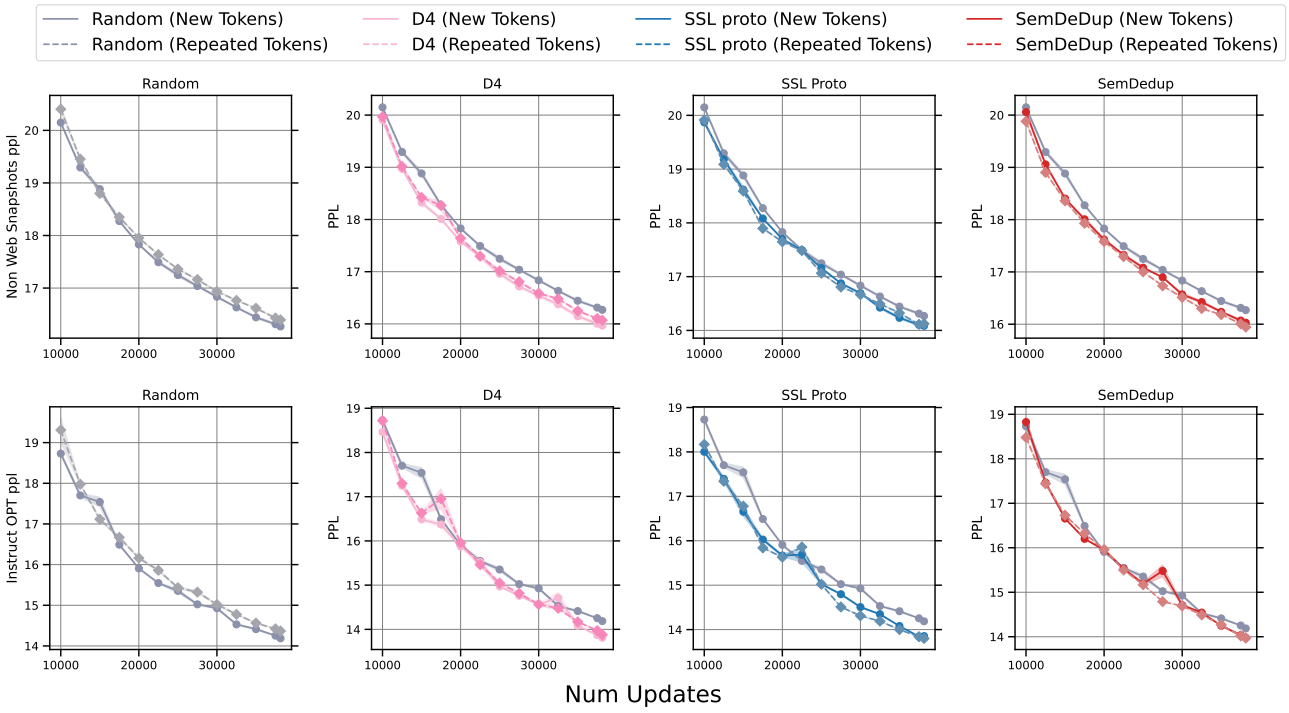

Figure A9. Effect of repeating tokens across data selection methods over training. X-axis denotes the number of updates, and the y-axis denotes average perplexity across non-web-snapshot validation sets (top row) and Instruct OPT (bottom row). Each column in the plot above denotes a different data selection method. Within each column: (1) the gray line denotes baseline training, (2) the colored-dashed line denotes repeating tokens via the specified data selection method, and (3) the colored-solid line denotes selecting new tokens via the specified data selection method. Repeating data is generally worse than selecting new data for a *fixed data selection method* (e.g., fixed column).

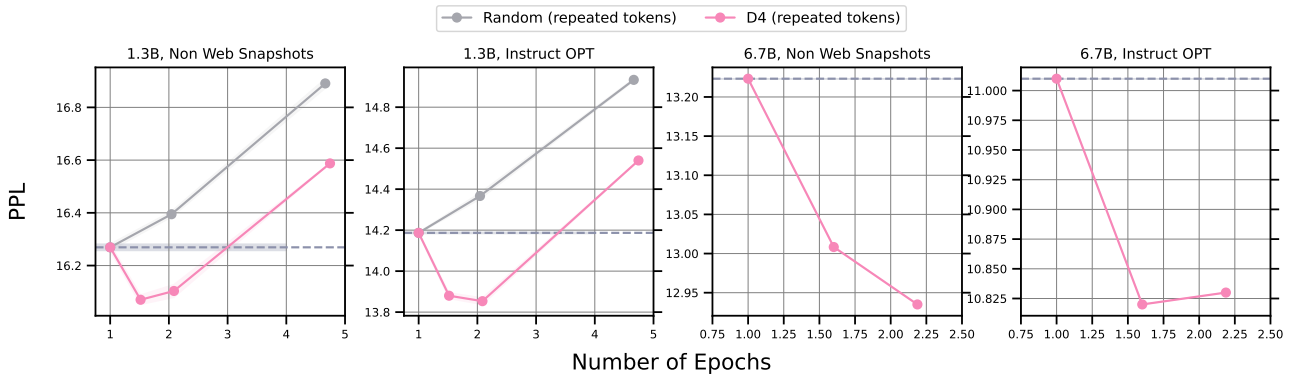

Figure A10. Comparison of repeating tokens with D4 (pink line), randomly selecting new tokens (horizontal dashed gray line), and randomly repeating data (gray line). We see with different epoch numbers. The y-axis denotes perplexity, and x-axis denotes number of epochs.

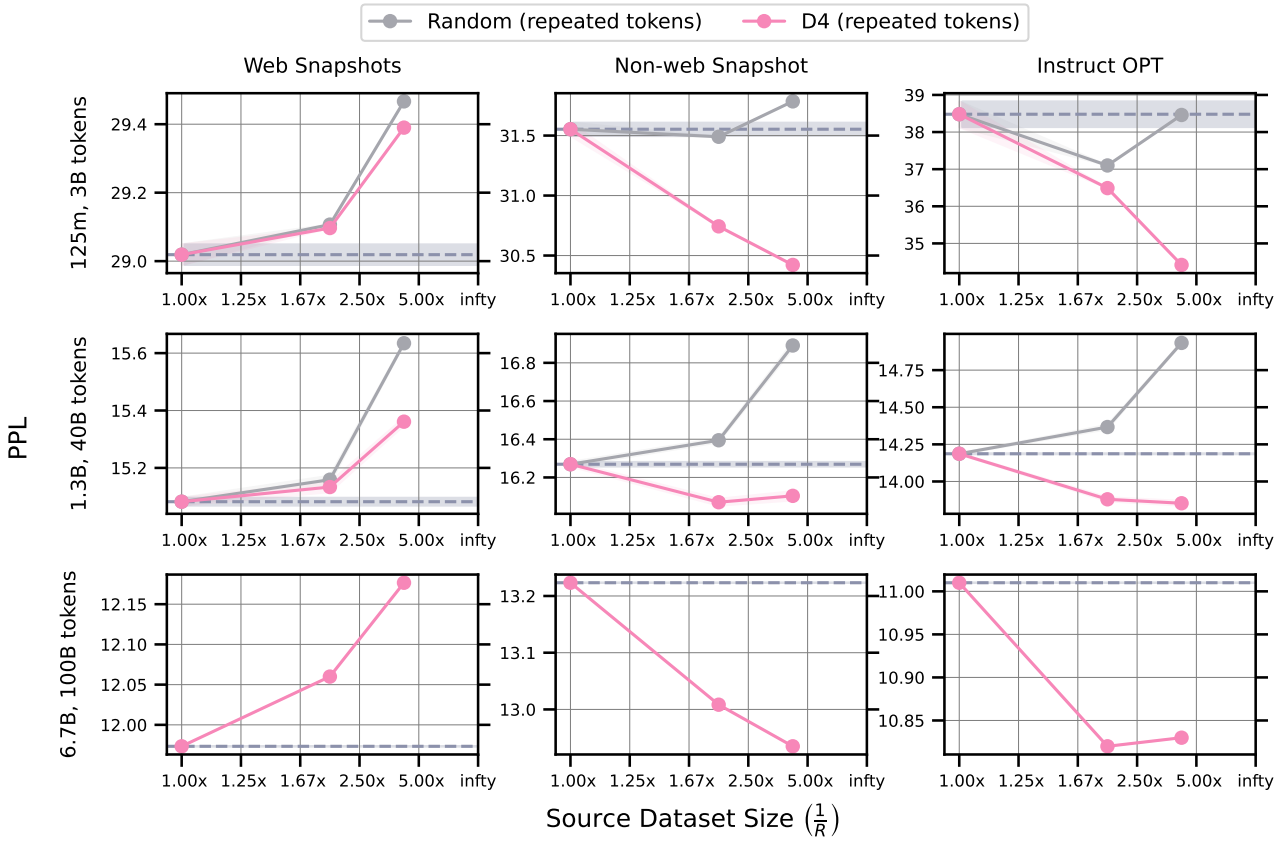

Figure A11. Comparison of repeating tokens with D4 (pink line), randomly selecting new tokens (horizontal dashed gray line), and randomly repeating data (gray line). We see across model scales (top: 125M trained on 3B tokens; middle: 1.3B trained on 40B tokens; bottom: 6.7B trained on 100B tokens) and data selection ratios, repeating data selected by D4 outperforms randomly selecting new data.