Research Article

Metadata Conditioning Accelerates Language Model Pre-training

Tianyu Gao[1], Alexander Wettig[1], Luxi He[1], Yihe Dong[1], Sadhika Malladi[1], Danqi Chen[1]

1. Princeton Language and Intelligence, Princeton University, United States

The vast diversity of styles, domains, and quality levels present in language model pre-training corpora is essential in developing general model capabilities, but efficiently learning and deploying the correct behaviors exemplified in each of these heterogeneous data sources is challenging. To address this, we propose a new method, termed Metadata Conditioning then Cooldown (MeCo), to incorporate additional learning cues during pre-training. MeCo first provides metadata (e.g., URLs like en.wikipedia.org) alongside the text during training and later uses a cooldown phase with only the standard text, thereby enabling the model to function normally even without metadata. MeCo significantly accelerates pre-training across different model scales (600M to 8B parameters) and training sources (C4, RefinedWeb, and DCLM). For instance, a 1.6B language model trained with MeCo matches the downstream task performance of standard pre-training while using 33% less data. Additionally, MeCo enables us to steer language models by conditioning the inference prompt on either real or fabricated metadata that encodes the desired properties of the output: for example, prepending wikipedia.org to reduce harmful generations or factquizmaster.com (fabricated) to improve common knowledge task performance. We also demonstrate that MeCo is compatible with different types of metadata, such as model-generated topics. MeCo is remarkably simple, adds no computational overhead, and demonstrates promise in producing more capable and steerable language models.[1]

Corresponding author: Tianyu Gao, tianyug@princeton.edu

1. Introduction

Language models (LMs) achieve remarkable general-purpose capabilities by training on vast web-sourced corpora. This diversity in training data underscores a fundamental challenge: while humans

naturally calibrate their understanding based on the source of the data, LMs process all content as equivalent samples. For instance, Internet documents about Apple CEO Tim Cook range from memes ("*Tim doesn't cook anymore*") to biographies ("*Tim Cook is the CEO of Apple*"). Treating data from these heterogeneous sources identically causes two issues: (1) it overlooks crucial contextual signals that aid comprehension, and (2) it can impede models from reliably surfacing appropriate behaviors (e.g., humor or factuality) for specialized downstream tasks.

To provide additional information about each document's source, we propose conditioning documents with their corresponding metadata during pre-training by prepending the widely available source URLs to each document. For instance, as shown in Figure 1, adding the source URLs to Tim Cook documents helps the model distinguish among a meme, a biography, an interview article, and a financial report. To ensure the model operates effectively with or without metadata during inference, we implement a "cooldown" phase for the final 10% of training, during which we train on standard data without metadata. We call this pre-training method **Me**tadata **Co**nditioning then **Co**oldown (MeCo).

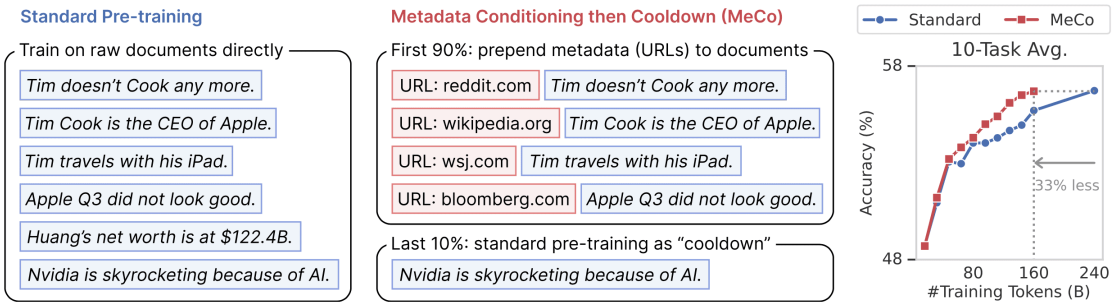

Figure 1. A comparison between data used by standard pre-training and MeCo. The figure on the right demonstrates 5-shot downstream task performance averaged across 10 tasks (1.6B models; details about the experiments can be found in §3).

Although metadata conditioning has been used in prior works to steer model generations[1] and improve the robustness of models to malicious prompts[2], our work establishes the general-purpose utility of this method in two crucial ways. First, we demonstrate that this paradigm can directly accelerate language model pre-training and improve downstream performance. Second, the cooldown phase in MeCo ensures the model can perform inference without metadata, unlike previous methods. We outline our specific contributions below.

1. **MeCo substantially accelerates pre-training (§3).** We demonstrate that MeCo enables a 1.6B model to achieve the same average downstream performance as a standard pre-trained model using 33% less training data. MeCo exhibits consistent gains across model scales (600M, 1.6B, 3B, and 8B) and data sources (C4, RefinedWeb, and DCLM).
2. **MeCo unlocks a new way to steer language models (§4).** Prepending appropriate real or synthetic URLs to the prompt during inference can induce desired model behaviors. For example, using factquizmaster.com (not a real URL) can enhance performance on common knowledge tasks (e.g., a 6% absolute improvement on zero-shot commonsense question answering), and using wikipedia.org reduces the likelihood of toxic generations several-fold compared to the standard unconditional inference.
3. **We ablate the design choices for MeCo (§5.1) and demonstrate that MeCo is compatible with different types of metadata (§5.2).** Ablations using hashed URLs and model-generated topics demonstrate that the main role of the metadata is to group documents together by source. As such, MeCo can effectively incorporate different types of metadata, including more fine-grained options, even when URLs are not available.

Our findings demonstrate that MeCo can significantly improve the data efficiency of language models while adding negligible computational overhead and complexity to the pre-training procedure. Moreover, the enhanced steerability afforded by MeCo holds promise in creating more controllable language models, and its general compatibility with more fine-grained and creative metadata warrants further exploration. Altogether, MeCo is a simple, flexible, and effective training paradigm that can simultaneously improve the utility and steerability of language models.

2. MeCo: Metadata Conditioning then Cooldown

In this section, we describe our pre-training approach in details. We assume each document in the pre-training dataset is associated with some metadata c. In our main experiments, we use the document URL's absolute domain name as c. For example, if the document's URL is https://en.wikipedia.org/wiki/Bill_Gates, then c is en.wikipedia.org (please refer to §5.2 for ablations on other URL variants). This URL information is readily available in many pre-training corpora, since most of them are derived from CommonCrawl[2], an open repository of web-crawled data.

Our method consists of two training stages, as illustrated in Figure 1.

1. **Pre-training with metadata conditioning (first 90%):** The model is trained on a concatenation of the metadata and the document, following this template: `URL: en.wikipedia.org\n\n[document]`. When using other types of metadata, URL should be replaced with the corresponding metadata name. **We only calculate the cross entropy loss over the document tokens**, disregarding those from the template or the metadata, as we found in our preliminary experiments that training on those tokens slightly hurts downstream performance.
2. **Cooldown with standard data (last 10%):** Models trained solely on metadata-augmented data degrade in performance when used without metadata (please refer to results in Table 4). To ensure general usage, we train the model on standard pre-training documents without any metadata during a cooldown stage, which covers the final 10% of steps in the pre-training process. The cooldown stage inherits the learning rate schedule and optimizer states from the metadata conditioning stage—i.e., it initializes the learning rate, model parameters, and optimizer states from the last checkpoint of the previous stage and continues adjusting the learning rate according to the schedule. Please refer to §A.3 for more details.

We also employ the following techniques in all our experiments, as they enhance the baseline pre-trained models' performance based on our preliminary experiments: (1) we disable cross-document attention[3][4], which both speeds up the training (25% faster for a 1.6B model) and improves the downstream performance (§B.1); (2) when packing multiple documents into one sequence, we ensure each sequence starts with a new document rather than in the middle of one—this may result in some data being discarded when packing documents to a fixed length, but it proves beneficial for improving downstream performance.

3. MeCo Improves Pre-training Data Efficiency

In this section, we demonstrate that MeCo can significantly accelerate language model pre-training (§3.2). We also show that MeCo leads to consistent gains across different model scales (§3.3) and training data (§3.4).

3.1. Experiment setup

We utilize the Llama[5][6][3] version of the Transformer architecture[7] and the Llama-3 tokenizer for all our experiments. We conduct experiments with four different model sizes: 600M, 1.6B, 3B, and 8B. The architecture details are in §A.2. We employ standard optimization settings for language models,

i.e., AdamW optimizer and cosine learning rate schedule. We follow[8] for hyperparameters and the details can be found in §A.1. Due to the high cost associated with pre-training and our limited resources, we perform only one run for each experiment; however, we demonstrate in §B.2 that the variance of our experiments should be low. §A.5 outlines the resources required for our experiments.

Pre-training data. We use the best-performing open-source pre-training corpus, DCLM-Baseline[8], for our main experiments. Additionally, we conduct experiments with two other data sources: a reproduction of RefinedWeb[9] from[8] and the C4 dataset[10]. In the paper, we refer to these data sources as DCLM, RefinedWeb, and C4, respectively. Notably, DCLM is a subset of RefinedWeb, acquired by using a fastText classifier[11] for selecting high-quality data[8]. Please refer to §A.4 for more details.

Evaluation. We adopt the OLMES suite[12] for evaluation, which includes the following tasks: MMLU[13], ARC-Easy (ARC-e;[14]), ARC-Challenge (ARC-c;[14]), CommonsenseQA (CSQA;[15]), HellaSwag (HSwag;[16]), OpenBookQA (OBQA;[17]), PIQA[18], Social IQA (SIQA;[19]), and WinoGrande (WG;[20]). We also add the popular TruthfulQA dataset (TruQA;[21]). Throughout the paper, we report the average performance across all 10 tasks as "Avg.". Unless specified, we always report 5-shot in-context learning results. OLMES enhances evaluation reliability by offering three key features: (1) it provides manually-curated in-context examples for each task; (2) it evaluates with both a multiple-choice format and a cloze format, and takes the best of two; (3) it applies ablated calibration method[22][23] to each individual task. During evaluation, we sample 1,000 examples for each task, which improves efficiency while providing the same reliable results as full evaluation.

3.2. MeCo achieves comparable performance to standard pre-training with 33% less data

Model	PPL	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
Standard	13.2	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
+ Data sel.	*13.3*	*37.2*	*74.6*	*44.3*	*62.9*	*65.5*	*46.8*	*74.3*	*52.4*	*64.3*	*37.8*	*56.0*
+ 80B tokens	*12.9*	*37.1*	*75.2*	*43.2*	*64.1*	*67.7*	*49.8*	*74.7*	*54.9*	*62.8*	*37.8*	*56.7*
MeCo	13.3	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	**56.7**
		↑0.2	↑0.6	↑1.4	↓1.0	↑0.6	↑5.2	↓0.9	↓1.6	↑2.2	↑3.3	↑1.0

Table 1. Our main experimental results of pre-training a 1.6B language model on 160B tokens from DCLM. MeCo significantly outperforms standard pre-training and achieves equivalent average performance to the 240B-token baseline while using 33% less data. Interestingly, validation perplexity (PPL) does not correlate with downstream performance.

Table 1 shows our main results of pre-training a 1.6B language model on 160B tokens from DCLM. Besides standard pre-training (**Standard**), we also feature two other experiments, both of which use more resources and only serve as references instead of fair comparisons:

- Data selection **(+ Data sel.)**: We employ the fastText data selection classifier from[8] to choose the top 70% documents from a 250B-token pool of DCLM data—this is similar to the high-quality data used in Section 5 of[8]. According to the Table 4 from[8], this fastText classifier achieves state-of-the-art data selection performance. This method incurs additional computational cost since the classifier must be applied over the whole corpus.
- Training with more data **(+ 80B tokens)**: We train a standard model with 240B tokens, with the same optimization hyperparameters.

We first observe that MeCo achieves significantly better performance than standard pre-training across most tasks. Additionally, MeCo surpasses the data selection baseline[3]; unlike data selection, our approach does not incur any computational overhead, as it leverages readily available URL information from the pre-training data. More importantly, MeCo achieves performance comparable

to standard pre-training while using 33% less data and compute, representing a substantial gain in data efficiency.

We also illustrate the changes in downstream task performance throughout the pre-training process in Figure 2. For MeCo, each checkpoint in the figure includes a cooldown phase on 16B tokens (10% of the total training tokens). For instance, the 80B checkpoint consists of 64B tokens of conditional training followed by 16B tokens of cooldown. We observe that MeCo consistently surpasses the baseline model, particularly in the later stage of training.

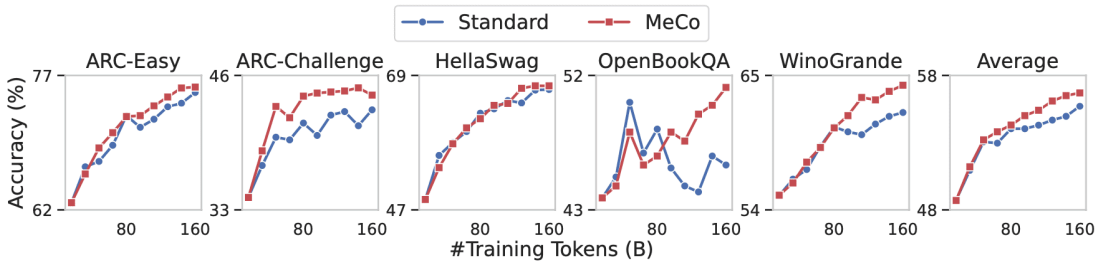

Figure 2. MeCo downstream task performance throughout training (1.6B model on DCLM). Each checkpoint of MeCo includes a 16B-token cooldown in the end. The total number of tokens used by the baseline and the corresponding MeCo checkpoints are the same for fair comparison. The reported average numbers are over all 10 tasks. Full results in Table 16.

Discussion of perplexity. Table 1 reveals that validation perplexity does not correlate with downstream performance in our experiments. Notably, when comparing the 240B baseline to the 160B MeCo model, the baseline exhibits much lower perplexity due to the larger data size, yet the two models achieve similar average downstream performance. This observation aligns with previous studies[24][25][26] indicating that perplexity is not always a reliable indicator of downstream performance; the final task performance can be impacted by other critical factors, such as inductive bias.

3.3. MeCo improves performance across model scales

Figure 3 demonstrates the results across different model scales (600M, 1.6B, 3B, and 8B). We train all the models with the same optimization hyperparameters and the same amount of data (160B on

DCLM) except for the 8B model, which is trained on 80B tokens with a lower learning rate due to resource constraints and training instability (details in §A.1).

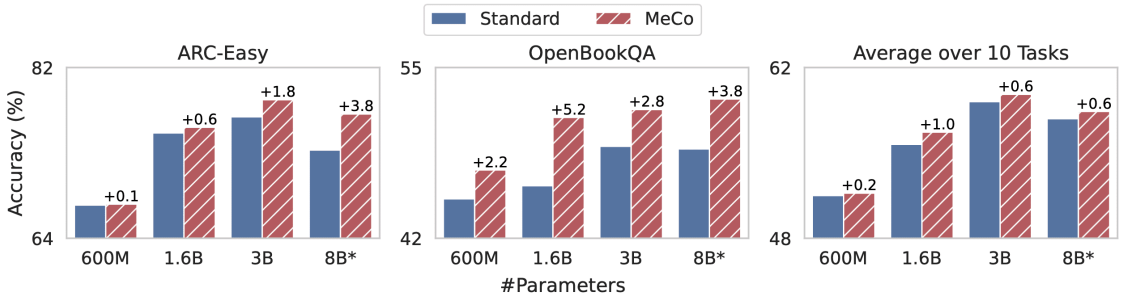

Figure 3. MeCo results across different model scales (160B tokens from DCLM except for the 8B* model, which is trained on 80B tokens due to resource constraints). Full results in Table 17. We report the average numbers across all 10 tasks. MeCo improves models across scales and leads to more gains for billion-parameter models compared to smaller models.

We first observe that MeCo improves model performance across all scales. Although the trend is somewhat noisy, MeCo appears to yield greater improvements for larger models, with billion-parameter models showing more significant gains compared to the 600M model. Note that this is a qualitative observation, as downstream task performance is known to scale less smoothly compared to pre-training loss.

3.4. MeCo improves performance across different training corpora

We train 1.6B models on 160B tokens from three different data sources: C4, RefinedWeb, and DCLM. We present the results in Figure 4. If we use the average downstream performance as an indicator for data quality, we can rank the three data sources as DCLM $>$ RefinedWeb $>$ C4. We observe that MeCo provides consistent and significant gains across different data sources, both on the average accuracies and individual tasks.

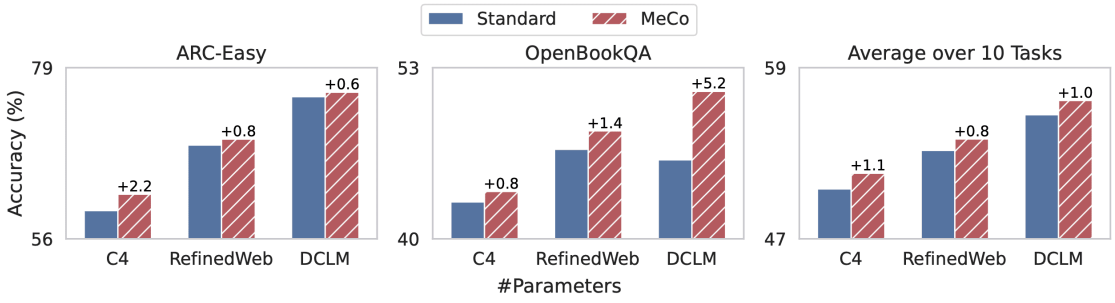

Figure 4. Results of applying MeCo over different pre-training corpora (1.6B models, 160B tokens). Full results in Table 18. We report the average numbers across all 10 tasks. MeCo provides consistent gains across different pre-training sources.

4. Conditional Inference Steers Language Model Generations

MeCo not only improves the general quality of pre-trained language models (evaluated by standard few-shot downstream task performance), but also unlocks the possibility of steering the model's generations during inference by conditioning it on particular URLs. We term this paradigm **conditional inference**, as illustrated in Figure 5.

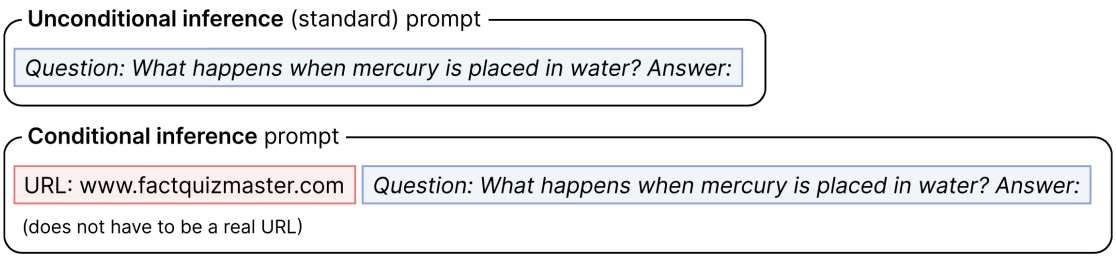

Figure 5. Illustration of conditional inference: We can condition the model by prepending a URL to the prompt. The URL does not need to be a real one.

Steering language model generations by conditioning the model on a "control sequence" has been explored in the past, either for style control[1] or for avoiding harmful content[2]. In this section, we study how combining conditional inference and MeCo (even with cooldown) can both improve the downstream task performance and reduce the likelihood of harmful generations.

4.1. Conditional inference improves MeCo's downstream task performance

In this section, we demonstrate how prepending appropriate URLs to the inputs improves MeCo's downstream performance. We first design a URL for each downstream task used in our evaluation, for example, www.factquizmaster.com for OpenBookQA and www.socialskillsassessment.com for Social IQA. You can find all the customized URLs in Table 11. We note that (1) the URLs do not need to be real; (2) we did not use trial-and-error when choosing the URLs to avoid overfitting to the test set.

We apply the same set of customized URLs to both the standard model and MeCo (1.6B, 160B DCLM tokens) and the results are shown in Table 2. We see that applying conditional inference leads to little difference on the standard model but a significant improvement on MeCo. Overall, MeCo with conditional inference achieves 1.5% absolute improvement compared to standard pre-training with unconditional inference.

Inference	**Pre-training**	
	Standard	**MeCo**
Unconditional	55.7	56.7
Conditional	55.8 ↑0.1	57.2 ↑0.5

Table 2. Conditional inference further improves MeCo performance (Table 19).

We also explore the impact of different URLs on performance, as shown in Table 3. In this experiment, we use two real URLs: boards.4chan.org, an anonymous imageboard known for its association with offensive content, and www.factmonster.com, a trivia website. Unlike our main experiment, we employ zero-shot prompting to highlight the effects of different URLs. Our findings indicate that selecting an appropriate URL can significantly enhance zero-shot results compared to using a more adversarial one: for example, using factmonster.com outperforms 4chan.org by 7.3% on CommonsenseQA.

Inference URLs	ARC-e	ARC-c	CSQA	OBQA
Unconditional inference	69.6	43.2	54.7	48.4
boards.4chan.org	66.7 ↓2.9	41.1 ↓2.1	53.6 ↓1.1	47.8 ↓0.6
www.factmonster.com	70.7 ↑1.1	45.7 ↑2.5	60.9 ↑6.2	52.4 ↑4.0

Table 3. Zero-shot evaluation of MeCo (1.6B, 160B DCLM tokens) with different URLs. We show the delta between unconditional inference and using URLs.

4.2. MeCo with conditional inference reduces harmful generations

In addition to improving downstream task performance, MeCo with conditional inference also reduces harmful generations. To evaluate the toxicity of model generations, we follow [27] to sample 4096 text sequences from the models, with temperature $T = 0.7$ and top-p=0.9. The generated sequences have lengths between 10 and 128 tokens. For unconditional inference, the model is only conditioned on the BOS token. For conditional inference, the model is conditioned on en.wikipedia.org.

To obtain toxicity scores, we follow the setup in [27] and use the toxic comment classifier Detoxify [28]. We use the unbiased model from Detoxify, which is based on RoBERTa [29] and trained on a human-labeled dataset of nearly 2 million comments, created for the task of evaluating unintended bias [30]. The classifier provides both general toxicity scores and more granular scores (e.g., obscene, insult).

We show the averaged toxicity scores over all sampled generations in Figure 6. We observe that using en.wikipedia.orgfor conditional inference reduces the toxicity scores of generations from both the standard pre-training model and MeCo. Conditional inference is more effective on MeCo, leading to a significantly lower toxicity score compared to the baseline.

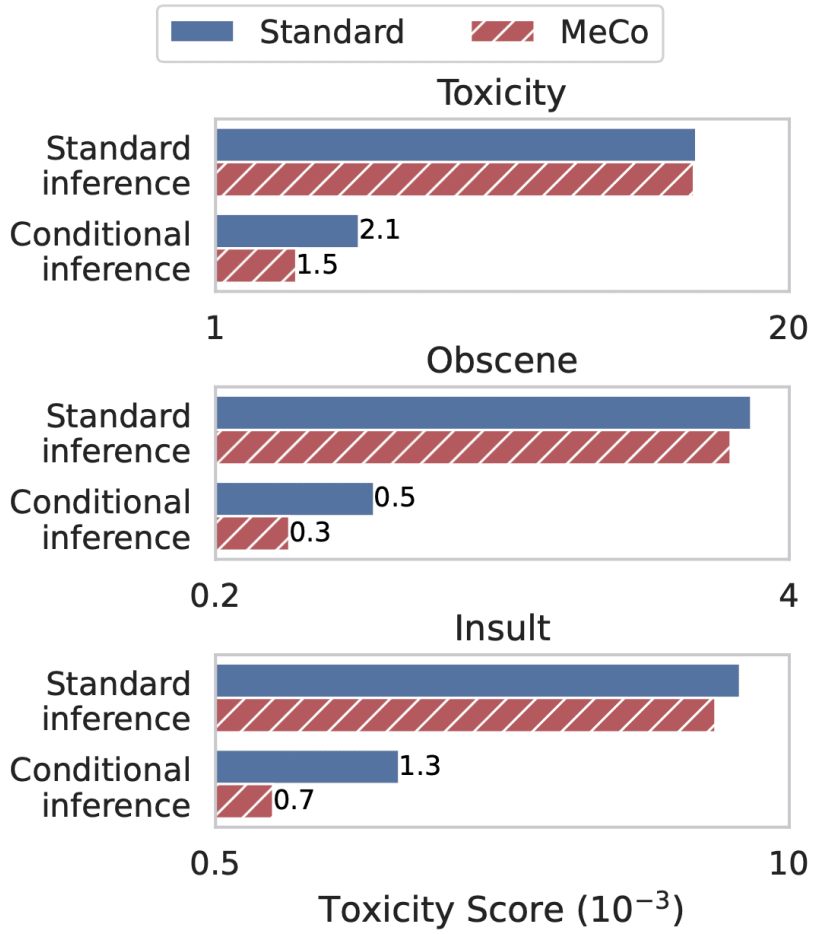

Figure 6. MeCo with conditional inference (using en.wikipedia.org) significantly reduces harmful generations.

5. Ablation Studies

5.1. Different strategies for mixing metadata-conditioned and standard data

In this section, we study the best strategy to mix metadata-augmented data and standard data. We experiment with four different strategies: only standard data, only metadata-conditioned data, directly mixing the two sources of data throughout training (90% URL + 10% standard) and two-stage training (i.e., first 90% with metadata conditioning and then 10% standard data)—the last one is MeCo.

Table 4 demonstrates the results of the different mixing strategies. First, we see that only training on metadata-conditioned data leads to performance degradation, emphasizing the importance of cooldown. While both directly mixing the two types of data and two-stage training improve the performance compared to the standard pre-training baseline, first training on metadata-conditioned data and then cooldown with standard data leads to better and more consistent gains. We also perform additional ablations on the length of cooldown in §B.3, which show that 10%-20% cooldown achieves the best performance (and we use 10% in our experiments).

Model	ARC-e	ARC-c	HSwag	OBQA	10-Task Avg.
100% standard	75.1	42.7	66.7	46.0	55.7
100% URL	72.4 ↓2.7	28.8 ↓13.9	61.5 ↓5.2	42.6 ↓3.4	50.3 ↓5.4
90% URL + 10% standard	72.5 ↓2.6	43.1 ↑0.4	66.9 ↑0.2	50.0 ↑4.0	56.4 ↑0.7
MeCo	**75.7**↑0.6	**44.1**↑1.4	**67.3**↑0.6	**51.2**↑5.2	**56.7**↑1.0

Table 4. Different strategies of mixing metadata-augmented and standard data. Full results can be found in Table 20.

5.2. Understanding the role of metadata

To better understand how MeCo works, we experiment with various types of metadata and present the results in Table 5. Below, we describe these metadata types and their outcomes.

Metadata	Examples	Avg.
URLs (MeCo)	en.wikipedia.org	56.7
Full URLs	en.wikipedia.org/wiki/Bill_Gates	56.8 ↑ 0.1
URL suffixes	org	56.2 ↓ 0.6
Top 0.2% URLs	en.wikipedia.org or unknown	56.4 ↓ 0.3
Top 2% URLs	en.wikipedia.org or unknown	56.3 ↓ 0.4
Hashed URLs	7dsjuj3a-olp0	56.7 ↑ 0.0
Model-generated topics	Technology leader biography	56.6 ↓ 0.1

Table 5. Ablations on using different metadata for MeCo. The average results are over all 10 tasks. Full results can be found in Table 21.

URL variants. We test URL variants that provide more information (full URLs) and less information (URL suffixes). While full URLs perform similarly to MeCo, using URL suffixes results in significant performance degradation, suggesting that absolute domain names (e.g., en.wikipedia.org) provide the appropriate granularity as metadata.

Top URLs. We retain only the most frequently appearing URLs from the DCLM data and mark others as "unknown". We experiment with two tiers: top 0.2% URLs (each URL corresponds to roughly more than 1,000 documents, covering 41.6% of the DCLM data) and top 2% URLs (each URL corresponds to more than 100 documents, covering 65.1% of the DCLM data). The URL distribution in DCLM is highly skewed, with a few top URLs covering a large portion of the data. Examples of top URLs are shown in Table 15. This experiment aims to determine whether MeCo primarily benefits from modeling infrequent or high-frequency URLs. We find that using only top URLs does not match MeCo's performance, indicating that MeCo also benefits from low-frequency URLs.

Hashed URLs. We map each unique URL into a random string to investigate whether MeCo needs to learn the semantics of URLs or simply recognizes that certain documents belong to the same groups. Surprisingly, using hashed URLs achieves performance on par with semantically-meaningful URLs,

indicating that the semantic meaning of the metadata is not necessary for better pre-trained models —instead, simply providing signals that group certain documents together is sufficient for improving pre-training data efficiency.

Model-generated topics. We explore ways of generating metadata in case readily available metadata is absent or insufficient. We prompt a Llama-3.1-8B-Instruct model to generate a two-word or three-word topic for each document, such as "technology leader biography" or "gaming forum" (more details in §A.6). This is more fine-grained metadata compared to domains (e.g., "Wikipedia" or "Books"). Note that prompting models to generate topics is extremely expensive, taking roughly 1,500 GPU hours, similar to what is required to pre-train the 1.6B model. Hence, it is not a practical method but included for analysis purposes. We observe that using model-generated topics leads to similar results to our main MeCo model, suggesting that metadata based on document contents instead of sources is equally useful, prompting future explorations on more creative ways of generating metadata.

Our ablations suggest that metadata conditioning improves pre-training data efficiency by grouping documents together by source or topic. We propose two preliminary hypotheses as to how metadata conditioning affects model training: First, the model may automatically learn to prioritize documents from useful sources or topics, thereby internally optimizing the mixture of training domains, which has been shown to be useful during pre-training[31][32]. Indeed,[33] also suggested that language models may autonomously identify domains rich in knowledge. Second, the model may use the additional metadata supervision to simply learn more structured representations of these large corpora, with no knowledge of the quality of each of the groups. We believe that the precise mechanism by which MeCo accelerates pre-training and improves model steerability warrants further theoretical and empirical study.

6. Related Work

Metadata conditioning. CTRL[1] first proposed "conditional language models" for controlled generation: the method prepended the pre-training documents with "control codes" such as source domains, which allowed for steering the generation during inference by prompting the model with different control codes.[34] used timestamps as the metadata to train time-aware language models and[35] adopted document languages as the metadata for a multilingual pre-trained model.[36] pre-trained language models on hyper text, which provided extra metadata such as class and id, which

allowed for conditional inference as well. Conditional training was also explored in alignment and preference optimization:[2] pre-trained models with reward model scores as the prefix and[37] [38] conditioned the text on their quality measurements in post-training—both allowed prompting the model with a high quality score during inference to output more human-preferred text. Besides, [39] used a similar idea to inject "document IDs" into the pre-training corpus to enable training data attribution, though the "IDs" were appended, instead of prepended to the documents. Recently, [33] found in a synthetic experiment setting that prepending a special token to distinguish knowledge-intensive documents (for example, documents from Wikipedia) from other ones improved the model's capacity to memorize knowledge.

While our findings on improved steerability after conditional training echo previous literature, our paper is the first to explore the use of metadata conditioning in modern-scale language model pre-training and its effect on downstream task performance. Compared to other types of metadata explored by prior work, we use URLs as they can be acquired with no additional cost and they are more informative than source domains or reward scores.

Selecting pre-training data. The quality of pre-training corpora is essential for the performance of the resulting language models. Consequently, there has been a huge amount of effort invested into improving pre-training data, starting from heuristic-based filtering[10][40][41][9][42] and deduplication[43][44][5][45]. Recently, model-based data filtering or data selection has emerged: many works sought to use simple ngram models to select those that resemble high-quality domains such as Wikipedia[22][46][8] or to use an existing language model for perplexity filtering[47][48][49]. [50][26][51] [3] instead used a large language model to score instances based on abstract values such as whether they are "educational"—but these methods introduce considerable overheads as running these language models over the whole pre-training corpus is costly and whether they can lead to better performance under the same computational budget is unclear[52][53].

Another line of works aimed to adjust the domain mixture for more data-efficient training[31][54][32]. However, these models require an existing domain taxonomy (which is usually very coarse-grained) and a target loss to optimize for—which has been shown to not always correlate with downstream performance[24][25].

7. Conclusion

We introduce metadata conditioning then cooldown (MeCo), an extremely simple method that consistently outperforms standard pre-training with negligible computational overhead. MeCo leverages commonly available metadata, such as source URLs, by prepending them to pre-training documents. At the end of training, MeCo removes the URLs from the data to enable inference without metadata. Through comprehensive experiments across various model scales and training corpora, we demonstrate MeCo's effectiveness, achieving up to a 33% speedup in pre-training. Additionally, we show that prompting MeCo models with suitable metadata can further enhance their downstream performance and mitigate harmful outputs. Our findings underscore the potential of metadata conditioning to enhance data efficiency in pre-training and to develop more controllable and steerable language models.

Limitations

Due to limited resources and the costly nature of pre-training, we do not perform multi-run experiments; however, we show in §B.2 that the variance of our experiments should be low and our results are significant. All our investigations are limited to English corpora. We do not study the interplay between metadata conditioning and post-training procedures. We also do not have a mechanistic understanding of how conditioning on metadata helps improve the downstream performance. We hope our results can shed light on these interesting questions and motivate further research on metadata conditioning.

Acknowledgments

We acknowledge Angelica Chen, Sanjeev Arora, Kyunghyun Cho, Yisong Yue, Luca Soldaini, and members of Princeton Language and Intelligence for their helpful feedback and discussion. Tianyu Gao is supported by an IBM PhD Fellowship. This research is funded by the National Science Foundation (IIS-2211779) and a Sloan Research Fellowship.

Appendix A. Experiment Details

A.1. Hyperparameters

Table 6 shows the hyperparameter settings used in our experiments. We follow[8] for the high learning rate and weight decay except for the 8B model, which requires a lower learning rate for numerical stability.

Hyperparameters	Values
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$)
Learning rate	3e-3 (5e-4 for the 8B model)
Weight decay	0.033 (0.1 for the 8B model)
Batch size	4M tokens
Warmup	5% linear warmup
Schedule	Cosine decay to 10% of the peak learning rate

Table 6. Hyperparameter settings for our experiments.

A.2. Model configurations

We use the Llama variant[5] of Transformers[7] for our experiments. All models use the Llama-3 tokenizer[3]. The detailed configurations are specified in Table 7.

#Param	#Layers	Hidden	Intermediate	#Heads	Head Dim
600M	24	1024	4096	16	64
1.6B	24	2048	5504	16	128
3B	28	3072	8192	24	128
8B	32	4096	14336	32	128

Table 7. Model configurations for our experiments.

A.3. Cooldown details

The metadata conditioning stage (90%) and the cooldown stage (10%) share the same learning rate schedule—i.e., the metadata conditioning stage will end at the 90% of the learning rate schedule and the cooldown stage will resume from that same point on the schedule and continue the learning rate decay. It also inherits all the optimizer states. To ensure the cooldown stage does not see repeated data as the conditional training stage, we use a different subset of data for cooldown for all our DCLM experiments.

For our 8B experiments (80B tokens), due to the checkpoint saving configuration, we performed a 10B-token cooldown (12.5% instead of 10% of the total training).

A.4. Dataset details

Table 8 shows the dataset details for our pre-training experiments.

Dataset	Description
C4	The SlimPajama[55] C4 subset
RefinedWeb	DCLM-reproduced[8] RefinedWeb
DCLM	DCLM-Baseline, which is a filtered version of DCLM-reproduced RefinedWeb

Table 8. Pre-training dataset details.

A.5. Experimental resource

Table 9 shows the resources required to train the models in our experiments. Our main models (1.6B, 160B tokens) take roughly 2 days to train on 32 H100 GPUs.

#Params	**600M**	**1.6B**	**1.6B**	**3B**	**8B**
#Tokens	160B	160B	240B	160B	80B
#GPU hours	776	1536	2304	3085	3905

Table 9. Resources required to train the models in our experiments (H100 GPU hours).

A.6. Prompts for model-generated topics

Table 10 shows the prompt used for generating topics. We prompt a Llama-3.1-8B-Instruct model to generate topics. We only use the first 1024 tokens from the document as the snippet. We use greedy decoding.

`Based on the given sampled snippet from a document (could be a webpage, a book, a codebase, a paper, or anything else), write a domain keyphrase (within 4 words; for example, code, international news, food blog, biography, science fiction, politics essay, gaming forum, algebra quiz, physics textbook, restaurant advertisement, religous story, etc.) for the document. The "domain keyphrase" should consider both the topics and the genre/source of the document.`
`*** Start of the snippet ***`
`{{snippet}}`
`*** End of the snippet ***`
`Now output the domain (do not output other things):`

Table 10. The prompt for generating topics.

A.7. Customized URLs for conditional inference

Table 11 shows the customized URLs for conditional inference.

Tasks	Customized URLs
MMLU	www.testprepportal.com
ARC-Easy	www.sciencestudyquiz.com
ARC-Challenge	www.sciencestudyquiz.com
CommonsenseQA	www.quizsmart.com
HellaSwag	www.wikihowquiz.com
OpenBookQA	www.factquizmaster.com
PIQA	www.basicknowledgequiz.com
Social IQA	www.socialskillsassessment.com
WinoGrande	www.testpreppractice.com
TruthfulQA	www.factcheckfun.com

Table 11. Customized URLs for conditional inference.

Appendix B. Additional Experiments

B.1. Cross-document attention ablation

Table 12 shows a comparison between enabling and disabling cross-document attention. Disabling cross-document attention leads to significant speedups for our training (for a 1.6B model, it is 25% faster). We also see that it brings a considerable performance improvement on the vanilla model. Interestingly, the average performance does not differ much between two different attention patterns for MeCo, suggesting that prepending the URLs to the document helps the model learn the noisy cross-document attention. Based on these results, all other experiments in this paper disable cross-document attention.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
Standard	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
+Cross-doc attn	36.3	73.4	41.6	63.2	65.5	46.0	73.6	52.4	61.3	36.7	55.0
MeCo	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	56.7
+Cross-doc attn	35.5	72.7	45.4	66.3	66.1	51.8	74.4	52.8	62.4	38.2	56.6

Table 12. Cross-document attention ablation (160B tokens, 1.6B parameters).

B.2. Experiment variance

Due to the nature of pre-training experiments and the high cost associated with it, we perform single runs for all our experiments and do not report their standard deviations. However, we provide a reference point here for estimating the variance of our experiments. We take the 90% checkpoint of the 1.6B-parameter, 160B-token standard pre-training model, and continue the rest 10% of the training with three disjoint sets of data. Table 13 shows their performance. We see that while some individual tasks show performance differences, the standard deviation of the average performance is very low (0.1%), demonstrating that the average performance across our selected tasks is an indicative and stable metric.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
Standard run 1	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
Standard run 2	36.2	73.9	43.4	63.1	67.5	46.2	74.2	53.2	62.0	35.5	55.5
Standard run 3	36.3	73.8	43.2	63.4	67.5	45.8	74.5	54.2	62.8	34.7	55.6
Avg.	36.2	74.3	43.1	63.8	67.2	46.0	74.3	53.9	62.3	35.1	55.6
Std.	±0.1	±0.7	±0.4	±0.9	±0.5	±0.2	±0.2	±0.6	±0.5	±0.4	±0.1

Table 13. Multiple runs of the baseline model (1.6B parameters, 160B tokens from DCLM). The average performance across runs shows low variance.

B.3. Cooldown length ablation

Table 14 shows the performance of different cooldown lengths. We see that performing a 10% and 20% cooldown achieves similar results, while further increasing the length hurts the performance. For simplicity, we use 10% cooldown for all our experiments. We note that the best cooldown length can vary across different numbers of parameters, total numbers of training tokens, and the pre-training corpora; however, performing such a fine-grained search across all different settings is intractable.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
10% cooldown	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	56.7
20% cooldown	36.5	74.7	46.0	64.2	67.1	49.4	73.6	53.3	64.3	39.0	56.8
30% cooldown	36.7	74.8	45.0	60.9	67.5	49.0	74.2	51.6	62.8	39.2	56.2

Table 14. Ablations on different cooldown lengths (1.6B parameters, 160B tokens).

Appendix C. DCLM URL Distributions

Table 15 shows the top 50 URLs from DCLM and the corresponding document ratios.

URLs	Document ratios
en.wikipedia.org	0.256%
stackoverflow.com	0.240%
www.theguardian.com	0.207%
www.urbandictionary.com	0.149%
www.fanfiction.net	0.148%
www.businessinsider.com	0.139%
gizmodo.com	0.123%
everything2.com	0.119%
www.physicsforums.com	0.100%
www.reference.com	0.090%
www.theatlantic.com	0.087%
www.mumsnet.com	0.086%
superuser.com	0.086%
chowhound.chow.com	0.085%
www.huffingtonpost.com	0.082%
serverfault.com	0.082%
www.engadget.com	0.079%
math.stackexchange.com	0.078%
www.nytimes.com	0.075%
news.bbc.co.uk	0.073%
gawker.com	0.071%
tvtropes.org	0.069%
www.instructables.com	0.069%
www.fool.com	0.068%
www.enotes.com	0.067%

URLs	Document ratios
townhall.com	0.067%
slashdot.org	0.066%
www.foxnews.com	0.066%
kotaku.com	0.066%
articles.chicagotribune.com	0.064%
www.reddit.com	0.063%
www.complex.com	0.063%
jezebel.com	0.062%
www.gamefaqs.com	0.061%
www.aljazeera.com	0.061%
askubuntu.com	0.061%
abcnews.go.com	0.060%
mathoverflow.net	0.058%
www.csmonitor.com	0.058%
articles.latimes.com	0.058%
www.bookrags.com	0.057%
lifehacker.com	0.057%
www.sfgate.com	0.057%
jalopnik.com	0.057%
www.ancestry.com	0.057%
www.nifty.org	0.057%
www.theregister.co.uk	0.057%
www.osnews.com	0.056%
www.cnet.com	0.055%
www.ign.com	0.055%

doi.org/10.32388/D226QH

Table 15. Top 50 URLs from DCLM.

Appendix D. Full Results

Table 16, Table 18, Table 17, Table 20, Table 21, and Table 19 show the detailed results of experiments reported in our main paper.

#Tokens	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
Standard											
16B	30.4	62.8	34.2	56.0	48.7	43.8	69.9	47.2	55.2	39.1	48.7
32B	32.1	66.8	37.3	60.0	55.9	45.2	70.3	46.7	56.5	38.6	50.9
48B	34.1	67.4	40.0	60.9	58.0	50.2	71.8	52.5	57.3	38.3	53.1
64B	34.0	69.2	39.8	61.6	59.8	46.8	72.7	50.2	59.2	36.3	53.0
80B	34.9	72.5	41.4	58.6	62.8	48.4	72.8	52.7	60.8	35.5	54.0
96B	34.9	71.2	40.2	62.1	63.5	45.8	72.4	53.5	60.4	36.4	54.0
112B	35.6	72.1	42.2	62.9	64.9	44.6	73.3	52.6	60.1	34.6	54.3
128B	35.9	73.5	42.5	62.8	64.5	44.2	73.1	53.9	61.0	35.3	54.7
144B	36.1	73.9	41.1	60.6	66.6	46.6	73.5	53.9	61.6	35.5	55.0
160B	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
MeCo											
16B	30.4	62.8	34.2	56.0	48.7	43.8	69.9	47.2	55.2	39.1	48.7
32B	32.5	66.0	38.7	58.2	53.9	44.6	70.6	49.4	56.2	41.8	51.2
48B	34.0	68.9	43.0	59.2	57.8	48.2	71.6	50.4	57.9	41.2	53.2
64B	34.2	70.6	41.9	62.6	60.4	46.0	72.1	50.5	59.1	40.1	53.8
80B	34.3	72.4	44.0	61.7	61.9	46.6	72.6	49.4	60.7	39.1	54.3
96B	34.9	72.5	44.3	63.1	64.1	48.2	72.9	49.5	61.7	38.7	55.0
112B	35.4	73.6	44.4	63.6	64.4	47.6	72.4	51.4	63.2	37.8	55.4
128B	35.7	74.6	44.5	64.9	66.9	49.4	73.0	51.5	63.0	37.5	56.1
144B	36.1	75.6	44.8	63.6	67.3	50.0	73.8	52.1	63.7	38.0	56.5
160B	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	56.7

Table 16. Intermediate checkpoint results for the 1.6B-parameter, 160B-token runs. For all MeCo checkpoints, we perform a 16B-token cooldown (i.e., the 64B checkpoint is 48B metadata conditioning

training + 16B cooldown).

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
600M model, 160B tokens from DCLM											
Standard	32.7	67.5	38.2	58.8	56.4	45.0	71.2	47.9	57.6	39.2	51.5
MeCo	32.8	67.6	37.0	62.0	54.2	47.2	71.0	49.6	57.1	37.9	**51.7**
1.6B model, 160B tokens from DCLM											
Standard	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
MeCo	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	**56.7**
3B model, 160B tokens from DCLM											
Standard	39.8	76.8	48.3	66.0	74.1	49.0	76.9	56.0	66.5	38.1	59.2
MeCo	39.7	78.6	48.5	71.0	73.6	51.8	77.0	55.5	65.9	36.4	**59.8**
8B model, 80B tokens from DCLM[†]											
Standard	39.2	73.3	46.0	66.0	72.8	48.8	76.1	54.8	66.2	35.2	57.8
MeCo	39.5	77.1	44.8	68.8	71.2	52.6	75.8	53.8	65.2	35.0	**58.4**

Table 17. Results with different numbers of parameters. All experiments use the same hyperparameters except for the 8B model[†], which uses a smaller learning rate and fewer tokens due to training instability and limited compute resources.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
1.6B model, 160B tokens from C4											
Standard	31.0	59.8	36.1	55.8	64.9	42.8	72.5	49.7	60.0	32.0	50.5
MeCo	31.9	62.0	37.8	54.3	63.6	43.6	74.0	50.0	58.9	39.5	**51.6**
1.6B model, 160B tokens from RefinedWeb											
Standard	32.4	68.6	37.1	61.2	63.9	46.8	73.9	51.2	59.7	36.7	53.2
MeCo	32.5	69.4	38.0	61.4	64.3	48.2	73.6	53.6	60.6	38.9	**54.0**
1.6B model, 160B tokens from DCLM											
Standard	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
MeCo	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	**56.7**

Table 18. Detailed results on different pre-training corpora.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
Standard	36.1	75.1	42.7	64.8	66.7	46.0	74.3	54.2	62.0	35.2	55.7
100% URL	33.9	72.4	28.8	37.2	61.5	42.6	72.9	52.1	60.5	41.0	50.3
90% URL + 10% Standard	36.4	72.5	43.1	63.7	66.9	50.0	75.7	53.1	62.8	39.9	56.4
MeCo	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	**56.7**

Table 20. Different strategies of mixing metadata-augmented and standard data.

Model	MMLU	ARC-e	ARC-c	CSQA	HSwag	OBQA	PIQA	SIQA	WG	TruQA	Avg.
URLs (MeCo)	36.3	75.7	44.1	63.8	67.3	51.2	73.4	52.6	64.2	38.5	56.7
Full URLs	36.7	75.4	43.9	68.3	66.5	51.2	74.0	52.9	63.2	35.6	56.8
URL suffix	36.2	73.9	42.7	65.2	67.7	49.0	73.1	53.6	62.1	38.1	56.2
Top 0.2% URLs	36.2	76.6	44.1	66.9	66.3	47.6	74.5	53.7	63.1	35.3	56.4
Top 2% URLs	36.5	73.5	44.8	65.4	65.8	48.2	74.3	53.4	64.3	36.9	56.3
Hashed URLs	36.4	73.7	44.2	64.6	67.2	51.8	74.3	54.8	62.5	37.9	56.7
Topics	36.3	74.5	45.3	64.5	67.4	48.2	74.2	53.5	63.1	38.6	56.6

Table 21. Experiment results on using different types of metadata.

Footnotes

[1] Our models, data, and code are available at https://github.com/princeton-pli/MeCo.

[2] https://commoncrawl.org/

[3] It is important to note that the DCLM data is already a subset of the RefinedWeb data selected by this classifier. We do not claim that MeCo consistently outperforms data selection; rather, we demonstrate that MeCo can be integrated with data selection to achieve further improvements, while data selection alone tends to yield diminishing returns.

References

1. a, b, c *Keskar NS, McCann B, Varshney LR, Xiong C, Socher R (2019). "Ctrl: A conditional transformer language model for controllable generation". arXiv preprint arXiv:1909.05858. Available from: https://arxiv.org/abs/1909.05858.*
2. a, b, c *Korbak T, Shi K, Chen A, Bhalerao RV, Buckley C, Phang J, Bowman SR, Perez E (2023). "Pretraining language models with human preferences". International Conference on Machine Learning. pp. 17506–17533.*

3. a, b, c, dDubey A, Jauhri A, Pandey A, Kadian A, Al-Dahle A, Letman A, Mathur A, Schelten A, Yang A, Fa n A, et al. The Llama 3 herd of models. arXiv preprint arXiv:2407.21783. 2024.
4. ^Ding H, Wang Z, Paolini G, Kumar V, Deoras A, Roth D, Soatto S. Fewer truncations improve language modeling. In: Forty-first International Conference on Machine Learning; 2024.
5. a, b, cTouvron H, Lavril T, Izacard G, Martinet X, Lachaux MA, Lacroix T, Rozière B, Goyal N, Hambro E, Azhar F, et al. LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.1397 1. 2023. Available from: https://arxiv.org/abs/2302.13971.
6. ^Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, Bashlykov N, Batra S, Bhargava P, Bhos ale S, Bikel D, Blecher L, Canton Ferrer C, Chen M, Cucurull G, Esiobu D, Fernandes J, Fu J, Fu W, Fuller B, Gao C, Goswami V, Goyal N, Hartshorn A, Hosseini S, Hou R, Inan H, Kardas M, Kerkez V, Khabsa M, Klo umann I, Korenev A, Koura PS, Lachaux MA, Lavril T, Lee J, Liskovich D, Lu Y, Mao Y, Martinet X, Mihay lov T, Mishra P, Molybog I, Nie Y, Poulton A, Reizenstein J, Rungta R, Saladi K, Schelten A, Silva R, Smith EM, Subramanian R, Tan XE, Tang B, Taylor R, Williams A, Kuan JX, Xu P, Yan Z, Zarov I, Zhang Y, Fan A, Kambadur M, Narang S, Rodriguez A, Stojnic R, Edunov S, Scialom T. "Llama 2: Open foundation and fine-tuned chat models, 2023."
7. a, bVaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017). "Atte ntion is all you need". Advances in Neural Information Processing Systems (NIPS). 30.
8. a, b, c, d, e, f, g, h, i, jLi J, Fang A, Smyrnis G, Ivgi M, Jordan M, Gadre S, Bansal H, Guha E, Keh S, Arora K, Garg S, Xin R, Muennighoff N, Heckel R, Mercat J, Chen M, Gururangan S, Wortsman M, Albalak A, Bitto n Y, Nezhurina M, Abbas A, Hsieh CY, Ghosh D, Gardner J, Kilian M, Zhang H, Shao R, Pratt S, Sanyal S, I lharco G, Daras G, Marathe K, Gokaslan A, Zhang J, Chandu K, Nguyen T, Vasiljevic I, Kakade S, Song S, Sanghavi S, Faghri F, Oh S, Zettlemoyer L, Lo K, El-Nouby A, Pouransari H, Toshev A, Wang S, Groeneve ld D, Soldaini L, Koh PW, Jitsev J, Kollar T, Dimakis AG, Carmon Y, Dave A, Schmidt L, Shankar V (2024). "Datacomp-lm: In search of the next generation of training sets for language models". arXiv preprint ar Xiv:2406.11794. Available from: https://arxiv.org/abs/2406.11794.
9. a, bGuilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessan dro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, Julien Launay. "The refinedweb dataset for falcon l lm: Outperforming curated corpora with web data only." In: A. Oh, T. Naumann, A. Globerson, K. Saenk o, M. Hardt, S. Levine, editors. Advances in Neural Information Processing Systems. Curran Associates, I nc.; 2023. p. 79155-79172.

10. ^ a, b *Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ (2020). "Exploring t he limits of transfer learning with a unified text-to-text transformer". The Journal of Machine Learning Research. 21 (1): 5485–5551.*
11. ^ *Joulin A, Grave E, Bojanowski P, Mikolov T (2017). "Bag of tricks for efficient text classification." In Pro ceedings of the 15th Conference of the European Chapter of the Association for Computational Linguisti cs: Volume 2, Short Papers, pp. 427–431.*
12. ^ *Gu Y, Tafjord O, Kuehl B, Haddad D, Dodge J, Hajishirzi H (2024). "Olmes: A standard for language mo del evaluations". arXiv preprint arXiv:2406.08446. Available from: https://arxiv.org/abs/2406.08446.*
13. ^ *Hendrycks D, Burns C, Basart S, Zou A, Mazeika M, Song D, Steinhardt J (2021). "Measuring massive m ultitask language understanding". Proceedings of the International Conference on Learning Representa tions (ICLR).*
14. ^ a, b *Clark P, Cowhey I, Etzioni O, Khot T, Sabharwal A, Schoenick C, Tafjord O (2018). "Think you have so lved question answering? Try ARC, the AI2 reasoning challenge". CoRR. arXiv:1803.05457.*
15. ^ *Talmor A, Herzig J, Lourie N, Berant J (2019). "CommonsenseQA: A question answering challenge targ eting commonsense knowledge". Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Sh ort Papers). pp. 4149–4158.*
16. ^ *Zellers R, Holtzman A, Bisk Y, Farhadi A, Choi Y (2019). "HellaSwag: Can a machine really finish your s entence?" Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 4791–4800.*
17. ^ *Mihaylov T, Clark P, Khot T, Sabharwal A (2018). "Can a suit of armor conduct electricity? A new datas et for open book question answering." Proceedings of the 2018 Conference on Empirical Methods in Nat ural Language Processing. pp. 2381–2391.*
18. ^ *Bisk Y, Zellers R, Le~bras R, Gao J, Choi Y (2020). "PIQA: Reasoning about physical commonsense in na tural language". Proceedings of the AAAI Conference on Artificial Intelligence. 34 (05): 7432–7439.*
19. ^ *Sap M, Rashkin H, Chen D, Le Bras R, Choi Y (2019). "Social IQa: Commonsense reasoning about social interactions". Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processin g and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 44 63–4473.*
20. ^ *Sakaguchi K, Le Bras R, Bhagavatula C, Choi Y (2021). "Winogrande: An adversarial winograd schema challenge at scale". Communications of the ACM. 64 (9): 99–106.*

21. ^Lin S, Hilton J, Evans O (2022). "TruthfulQA: Measuring how models mimic human falsehoods". Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 3214–3252.

22. a, bBrown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, et al. Language models are few-shot learners. In: Advances in Neural Information Processing Systems (NeurIPS); 2020.

23. ^Holtzman A, West P, Shwartz V, Choi Y, Zettlemoyer L (2021). "Surface form competition: Why the highest probability answer isn{'}t always right". Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 7038–7051.

24. a, bTay Y, Dehghani M, Rao J, Fedus W, Abnar S, Chung HW, Narang S, Yogatama D, Vaswani A, Metzler D. Scale efficiently: Insights from pretraining and finetuning transformers. In: International Conference on Learning Representations; 2022.

25. a, bLiu H, Xie SM, Li Z, Ma T (2023). "Same pre-training loss, better downstream: Implicit bias matters for language models". International Conference on Machine Learning. pp. 22188–22214. PMLR.

26. a, bWettig A, Gupta A, Malik S, Chen D. "QuRating: Selecting high-quality data for training language models." Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pp. 52915–52971, 21–27 Jul 2024.

27. a, bKorbak T, Shi K, Chen A, Bhalerao RV, Buckley C, Phang J, Bowman SR, Perez E. Pretraining language models with human preferences. In: Krause A, Brunskill E, Cho K, Engelhardt B, Sabato S, Scarlett J, editors. Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research. 2023 Jul 23-29;202:17506-17533. Available from: https://proceedings.mlr.press/v202/korbak23a.html.

28. ^Hanu L, Unitary team (2020). "Detoxify". Github. https://github.com/unitaryai/detoxify.

29. ^Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019). "RoBERTa: A robustly optimized BERT pretraining approach". arXiv preprint arXiv:1907.11692. Available from: https://arxiv.org/abs/1907.11692.

30. ^Borkan D, Dixon L, Sorensen J, Thain N, Vasserman L (2019). "Nuanced metrics for measuring unintended bias with real data for text classification". Companion Proceedings of The 2019 World Wide Web Conference, WWW '19, pp. 491–500, New York, NY, USA: Association for Computing Machinery.

31. a, bSang MX, Pham H, Dong X, Du N, Liu H, Lu Y, Liang PS, Le QV, Ma T, Yu AW (2024). "Doremi: Optimizing data mixtures speeds up language model pretraining". Advances in Neural Information Processing

Systems. 36.

32. a, b*Jiang Y, Zhou A, Feng Z, Malladi S, Kolter JZ (2024). "Adaptive data optimization: Dynamic sample s election with scaling laws". arXiv preprint arXiv:2410.11820. arXiv:2410.11820.*

33. a, b*Allen-Zhu Z, Li Y (2024). "Physics of language models: Part 3.3, knowledge capacity scaling laws". a rXiv preprint arXiv:2404.05405. Available from: arXiv:2404.05405.*

34. ^*Dhingra B, Cole JR, Eisenschlos JM, Gillick D, Eisenstein J, Cohen WW (2022). "Time-aware language models as temporal knowledge bases". Transactions of the Association for Computational Linguistics. 10: 257–273.*

35. ^*Liu Y, Gu J, Goyal N, Li X, Edunov S, Ghazvininejad M, Lewis M, Zettlemoyer L (2020). "Multilingual de noising pre-training for neural machine translation". Transactions of the Association for Computationa l Linguistics. 8: 726–742.*

36. ^*Aghajanyan A, Okhonko D, Lewis M, Joshi M, Xu H, Ghosh G, Zettlemoyer L (2022). "HTLM: Hyper-tex t pre-training and prompting of language models". International Conference on Learning Representati ons.*

37. ^*Lu X, Welleck S, Hessel J, Jiang L, Qin L, West P, Ammanabrolu P, Choi Y (2022). "Quark: Controllable t ext generation with reinforced unlearning". Advances in Neural Information Processing Systems. 35: 27 591–27609.*

38. ^*Liu H, Sferrazza C, Abbeel P. Chain of hindsight aligns language models with feedback. In: The Twelfth International Conference on Learning Representations; 2024.*

39. ^*Khalifa M, Wadden D, Strubell E, Lee H, Wang L, Beltagy I, Peng H. Source-aware training enables kno wledge attribution in language models. First Conference on Language Modeling. 2024.*

40. ^*Rae JW, Borgeaud S, Cai T, Millican K, Hoffmann J, Song F, Aslanides J, Henderson S, Ring R, Young S, et al. (2021). "Scaling language models: Methods, analysis & insights from training gopher". arXiv prep rint arXiv:2112.11446. Available from: https://arxiv.org/abs/2112.11446.*

41. ^*Laurençon H, Saulnier L, Wang T, Akiki C, Villanova del Moral A, Le Scao T, Von Werra L, Mou C, Gonz ález Ponferrada E, Nguyen H, Frohberg J, Šaško M, Lhoest Q, McMillan-Major A, Dupont G, Biderman S, Rogers A, Ben allal L, De Toni F, Pistilli G, Nguyen O, Nikpoor S, Masoud M, Colombo P, de la Rosa J, Vill egas P, Thrush T, Longpre S, Nagel S, Weber L, Romero Muñoz M, Zhu J, Van Strien D, Alyafeai Z, Almub arak K, Chien VM, Gonzalez-Dios I, Soroa A, Lo K, Dey M, Ortiz Suarez P, Gokaslan A, Bose S, Adelani DI, Phan L, Tran H, Yu I, Pai S, Chim J, Lepercq V, Ilic S, Mitchell M, Luccioni S, Jernite Y. The bigscience RO*

OTS corpus: A 1.6TB composite multilingual dataset. In: Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2022.

42. ^*Soldaini L, Kinney R, Bhagia A, Schwenk D, Atkinson D, Authur R, Bogin B, Chandu K, Dumas J, Elazar Y, Hofmann V, Jha A, Kumar S, Lucy L, Lyu X, Lambert N, Magnusson I, Morrison J, Muennighoff N, Naik A, Nam C, Peters M, Ravichander A, Richardson K, Shen Z, Strubell E, Subramani N, Tafjord O, Walsh E, Zettlemoyer L, Smith N, Hajishirzi H, Beltagy I, Groeneveld D, Dodge J, Lo K. Dolma: an open corpus of t hree trillion tokens for language model pretraining research. In: Ku LW, Martins A, Srikumar V, editors. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Lo ng Papers). 2024. p. 15725-15788.*

43. ^*Lee K, Ippolito D, Nystrom A, Zhang C, Eck D, Callison-Burch C, Carlini N (2022). "Deduplicating traini ng data makes language models better". Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 8424–8445.*

44. ^*Anil R, Dai AM, Firat O, Johnson M, Lepikhin D, Passos A, Shakeri S, Taropa E, Bailey P, Chen Z, et al. P aLM 2 technical report. arXiv preprint arXiv:2305.10403. 2023. Available from: https://arxiv.org/abs/23 05.10403.*

45. ^*Abbas A, Tirumala K, Simig D, Ganguli S, Morcos AS (2023). "{SemDeDup}: Data-efficient learning at web-scale through semantic deduplication". arXiv preprint arXiv:2303.09540. Available from: https://a rxiv.org/abs/2303.09540.*

46. ^*Sang MX, Santurkar S, Ma T, Liang P (2023). "Data selection for language models via importance resa mpling". Advances in Neural Information Processing Systems (NeurIPS).*

47. ^*Wenzek G, Lachaux MA, Conneau A, Chaudhary V, Guzm{\'a}n F, Joulin A, Grave E (2020). "CCNet: Ext racting high quality monolingual datasets from web crawl data". Proceedings of the Twelfth Language Resources and Evaluation Conference. pp. 4003–4012.*

48. ^*Muennighoff N, Rush AM, Barak B, Le Scao T, Tazi N, Piktus A, Pyysalo S, Wolf T, Raffel C. Scaling data -constrained language models. In: Thirty-seventh Conference on Neural Information Processing Syste ms; 2023.*

49. ^*Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker (2023). "W hen less is more: Investigating data pruning for pretraining LLMs at scale."*

50. ^*Gunasekar S, Zhang Y, Aneja J, Mendes CCT, Del Giorno A, Gopi S, Javaheripi M, Kauffmann P, de Rosa G, Saarikivi O, Salim A, Shah S, Behl HS, Wang X, Bubeck S, Eldan R, Kalai AT, Lee YT, Li Y (2023). "Text books are all you need."*

51. ^Guilherme Penedo, Hynek Kydl{\'\i}{\v{c}}ek, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von~Werra, Thomas Wolf, et~al. The fineweb datasets: Decanting the web for the finest text data at scale. arXiv preprint arXiv:2406.17557, 2024.

52. ^Goyal S, Maini P, Lipton ZC, Raghunathan A, Kolter JZ. Scaling laws for data filtering-- data curation cannot be compute agnostic. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2024. p. 22702–22711.

53. ^Kaddour J, Key O, Nawrot P, Minervini P, Kusner MJ (2024). "No train no gain: Revisiting efficient training algorithms for transformer-based language models". Advances in Neural Information Processing Systems. 36.

54. ^Xia M, Gao T, Zeng Z, Chen D. "Sheared LLaMA: Accelerating language model pre-training via structured pruning." In: The Twelfth International Conference on Learning Representations, 2024.

55. ^Soboleva D, Al-Khateeb F, Myers R, Steeves JR, Hestness J, Dey N (2023). "SlimPajama: A 627B token cleaned and deduplicated version of RedPajama".

Declarations

Funding: Tianyu Gao is supported by an IBM PhD Fellowship. This research is funded by the National Science Foundation (IIS-2211779) and a Sloan Research Fellowship.

Potential competing interests: No potential competing interests to declare.