Impact of Pretraining Term Frequencies on Few-Shot Numerical Reasoning

Yasaman Razeghi[◇] **Robert L. Logan IV**[♡] **Matt Gardner**[♠] **Sameer Singh**[◇♣]
[◇]University of California, Irvine [♡]Dataminr Inc.
[♠]Microsoft Semantic Machines [♣]Allen Institute for AI
{yrazeghi, sameer}@uci.edu
rlogan@dataminr.com mattgardner@microsoft.com

Abstract

Pretrained Language Models (LMs) have demonstrated ability to perform numerical reasoning by extrapolating from a few examples in few-shot settings. However, the extent to which this extrapolation relies on robust reasoning is unclear. In this paper, we investigate how well these models reason with terms that are less frequent in the pretraining data. In particular, we examine the correlations between the model performance on test instances and the frequency of terms from those instances in the pretraining data. We measure the strength of this correlation for a multiple GPT-based language models (pretrained on the Pile dataset) on various numerical deduction tasks (e.g., arithmetic and unit conversion). Our results consistently demonstrate that models are more accurate on instances whose terms are more prevalent, in some cases above 70% (absolute) more accurate on the top 10% frequent terms in comparison to the bottom 10%. Overall, although LMs appear successful at few-shot numerical reasoning, our results raise the question of how much models actually generalize beyond pretraining data, and we encourage researchers to take the pretraining data into account when interpreting evaluation results.

1 Introduction

Large language models have demonstrated outstanding zero- and few-shot performance on various reasoning benchmarks (Brown et al., 2020; Radford et al., 2019). In particular, their high performance on numerical tasks, such as addition and multiplication, suggests that models may have learned the ability to perform the underlying reasoning operations simply through a combination of pretraining and model size (Lewkowycz et al., 2022). As these numerical reasoning tasks become increasingly prevalent for evaluating the performance of

Q: What is 24 times 18? A: ___ *Model:* 432 ✓	Q: What is 23 times 18? A: ___ *Model:* 462 ✗

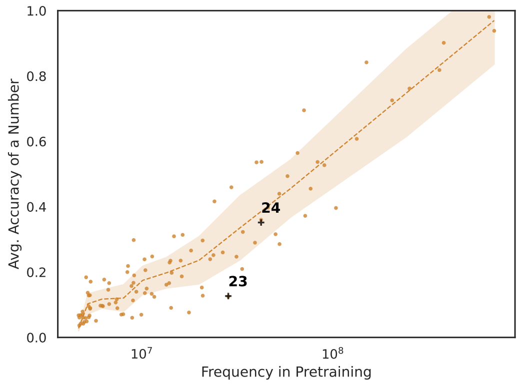


Figure 1: **Multiplication Performance:** Plot of GPT-J-6B's 2-shot accuracy on multiplication (averaged over multiple multiplicands and training instances) against the frequency of the equation's term in the pretraining corpus. Each point represents the average performance for that term (e.g., 24) multiplied by numbers 0-99 and 5 choices of random seeds. As in the example, the performance difference for the numbers 24 and 23 is more than 20%. We find a strong correlation between the accuracy for a number and its frequency in pretraining.

large language models (Chowdhery et al., 2022), it is crucial to understand the extent to which performance on these tasks reflects robust reasoning capabilities, especially since numerical reasoning is an essential skill needed to perform other complex reasoning tasks such as question answering through reading comprehension (Dua et al., 2019), and commonsense reasoning (Thawani et al., 2021; Lin et al., 2020a).

Current schemes for evaluating the reasoning of large language models, however, often neglect or underestimate the impact of data leakage from pretraining data. Although overlap between the training and evaluation splits of public datasets and its effect on the generalization of language models has been studied (Elangovan et al., 2021; Lewis et al., 2021a), the effect of the pretraining data has received less attention, and very few studies have at-

Work done while Robert Logan was at UCI.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 840 - 854

tempted to evaluate the effect of pretraining data on model's performance (Elazar et al., 2022). Ideally, a model that has *learned* to *reason* in the training phase should be able to generalize outside of the narrow context that it was trained in. Specifically, if the model has learned to reason numerically, its performance on instances with less frequent numbers (based on pretraining data) should not be significantly lower than its performance on the instances with common numbers.

For illustration, consider the arithmetic task of multiplying two integers (shown in Figure 1). A model that has learned proper arithmetic skills should be able to answer the queries irrespective of the frequencies of the operands in the pretraining data. Therefore, it should have roughly equivalent performance when answering the queries *Q: what is* 24 *times* X*?* and *Q: what is* 23 *times* X*?* despite the fact that 24 appears more frequently in the pretraining data. This is not the case with current LMs and we will study the effect of frequency terms in details through this paper. To show the effect of frequency, in this example, we plot the average accuracy of GPT-J-6B (Wang, 2021) on the numbers 0–99 (averaged over 0–99 as the other operand) against the frequency of the number in the pretraining data in Figure 1. We find a strong correlation between the term frequency and the model performance indicating that the model reasoning is not robust to these frequencies. Note that even "rare" terms still appear on the order of millions of times in the pretraining data.

In this work, we investigate this impact of the frequency of test instance terms in a model's pretraining data on the model's performance. We experiment on numerical reasoning tasks of addition, multiplication, and unit conversion. We count occurrences of the numbers and units in instances of these tasks in the pretraining data, including co-occurrences of term pairs or triples within a fixed window. This procedure allows us to aggregate over instances in which these terms appear and observe the relationship between term frequency and model accuracy on instances that include those terms. We summarize this behavior through the *performance gap* between instances that have the most frequent terms and instances that have the least frequent terms. Intuitively, models that exhibit a *high* performance gap are *much more* accurate on instances that are more common in the pretraining data, suggesting that the model *does not generalize* appropriately and is affected by dataset overlap.

We present analysis on these numerical reasoning tasks for three sizes of the EleutherAI/GPT models pretrained on the Pile dataset (Gao et al., 2020), which has been publicly released and thus permits this kind of analysis (in contrast to the data that, e.g., GPT-3 (Brown et al., 2020) was trained on). Our results consistently show a large performance gap between highest- and lowest-frequency terms; in some cases there is a more than 70% average accuracy gap between the top and bottom 10% terms. We also investigate whether this performance gap can be explained by strong memorization effects, i.e. by instances that are memorized by the language model. To achieve this, we remove instances that contain frequent *combinations* of numbers from our analysis, and study the performance on the remaining instances. Even in this case, we still find a strong correlation between frequency of terms and average performance, indicating that our results cannot be explained solely by direct memorization.

These observations suggest that any evaluation of reasoning that does not take the pretraining data into account is difficult to interpret, and that we need to revisit evaluation of language models with respect to their pretraining data before making any conclusion about the models generalization abilities beyond the pretraining data.

2 Background and Methodology

Numerical reasoning has been essential part of complex multi-step reasoning tasks for natural language understanding (Dua et al., 2019; Wei et al., 2022). Recently, large language models have exhibited an ability to perform numerical reasoning tasks in few-shot settings without requiring any modifications to their parameters through a method called in-context learning (Brown et al., 2020; Chowdhery et al., 2022). Our goal is to evaluate this reasoning skill in-depth and with respect to the *pretraining data*. This section provides background information on in-context learning and introduces our method for measuring the performance gap of the models on numerical reasoning tasks based on differences in pretraining term frequency.

The demonstration of all experiments in this paper is available at `https://nlp.ics.uci.edu/snoopy` (Razeghi et al., 2022) and the code is available at `https://github.com/yasamanrazeghi7/TermFrequency`

2.1 In-context Learning

Brown et al. (2020) show that the large GPT-3 model is able to perform well on few-shot reasoning tasks without requiring any changes to its internal parameters, through the usage of a technique called *in-context learning*. In place of a typical learning procedure, in-context learning instead places training examples in a prompt format, which is subsequently fed to a language model as its input. Recently, a few studies have researched the role of prompt and investigated the aspects that make in-context learning successful (Min et al., 2022; Zhao et al., 2021; Chan et al., 2022).

Among numerous experiments, Brown et al. (2020) show that GPT3 performs well on a variety of arithmetic questions such as addition and subtraction with 2–5 digit numbers. For example, they show that the largest model can perform zero-shot 2-digit addition with 76.9% accuracy. Although impressive, due to the large volume of data GPT-3 is trained on, it is possible that the model is repeating answers seen during pretraining. To attribute this performance to the model's *reasoning* capabilities, we need to make sure that the model is not affected by statistical overlaps between the terms of the arithmetic questions and the pretraining data.

In the following sections, we introduce metrics that we use to investigate the relationship between the frequency of terms in the pretraining data and the model performance on reasoning instances containing those terms. To assess this relation, we first define an approach for measuring term frequencies in a large pretraining dataset (Section 2.2). We connect these frequencies to reasoning performance by introducing the *performance gap* Δ (Section 2.3).

2.2 Frequency

We consider numerical reasoning tasks (Table 1) whose instances consist of input terms, $\mathbf{x} = (x_1, \ldots, x_i, \ldots x_n)$, and a derived output term y, where the x_i's are either positive integers or units of time (e.g., 1, 2, `hour`, etc.) and y is a positive integer. For example, for the task of multiplication, an instance might be $\mathbf{x} = (23, 18)$ and $y = 414$, representing the equation $23 \times 18 = 414$.

For each instance, we extract counts of the number of times that a subset of its terms $X \subseteq \{x_1, \ldots, x_n, y\}$ appear within a specified window in the pretraining data. We refer to this count as the frequency, ω_X, of X.

In this paper, we restrict our attention to frequencies involving three or less input terms, e.g., $\mathbf{x} = (x_1)$ or (x_1, x_2) or (x_1, x_2, x_3) and optionally the output term y, e.g.:

- $\omega_{\{x_1\}}$: the number of times that x_1 (one of the terms, e.g., 23) appears in the pretraining data.
- $\omega_{\{x_1,x_2\}}$: the number of times that the input terms x_1 (e.g., 23) and x_2 (e.g., 18) appear in the pretraining data within a specific window size.
- $\omega_{\{x_1,y\}}$: the number of times that the first input term x_1 (e.g., 23) and the output term y (e.g., 414) appear in the pretraining data within a specific window size.

Note that our usage of set notation in the subscript is deliberate; although $\mathbf{x} = (x_1, x_2)$ and $\mathbf{x}' = (x_2, x_1)$ are not necessarily the same (e.g., order is important when representing the task instance), frequency is symmetric (e.g., $\omega_{\{x_1,x_2\}} = \omega_{\{x_2,x_1\}} \forall x_1, x_2$).

2.3 Performance Gap

We want to measure how much *more* accurate the model is on instances containing more versus less frequent terms in the pretraining data. We do this by calculating the differences in average accuracies of the instances in the top and bottom quantiles of the distribution over term frequencies, which we call the *performance gap*.

Formally, let $\{(X^{(n)}, \omega_X^{(n)})\}$, $n \in [1, N]$, be a set of terms for a task and their associated term frequencies in the pretraining corpus. Given a task (e.g. addition), we create reasoning instances for each element of this set by instantiating values of x_i, and deriving y. We then measure the LM's accuracy $a^{(n)}$ over the set of instances, and repeat this process for all $n \in [1, N]$, producing a set $\Omega = \{(\omega_X^{(n)}, a^{(n)})\}$. The formula for the **performance gap** is then given by:

$$\Delta(\Omega) = \mathrm{Acc}(\Omega_{>90\%}) - \mathrm{Acc}(\Omega_{<10\%}) \quad (1)$$

where $\Omega_{>90\%}$ is the top 10% of elements in Ω ordered by frequency, $\Omega_{<10\%}$ is the bottom 10%, and $\mathrm{Acc}(\Omega')$ is the average accuracy of elements in Ω'. We introduce the following convenient abuses of notation Δ_1, $\Delta_{1,2}$, $\Delta_{1,y}$, $\ldots$, to denote the performance gap over the frequency distributions of $\omega_{\{x_1\}}, \omega_{\{x_1,x_2\}}, \omega_{\{x_1,y\}}, \ldots$, respectively.

Concretely, for the multiplication example from Figure 1, $\mathbf{x} = (x_1, x_2)$ and we consider the performance gap over frequencies $\omega_{\{x_1\}}$. For each number (say 23), we count the number of times it appears in the pretraining corpus ($\omega_{\{23\}}$), and

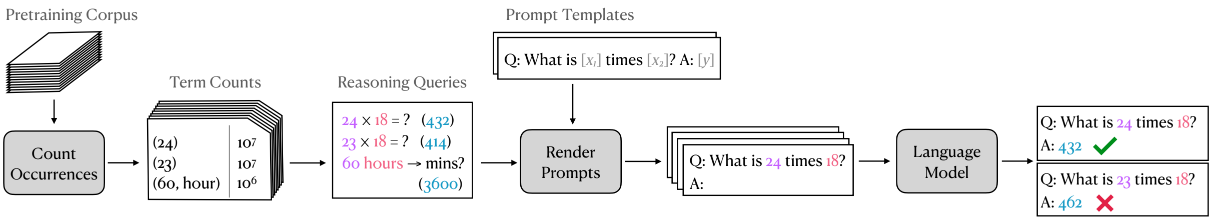


Figure 2: **Pipeline for Data Construction:** We use the term counts processed from the pretraining data to develop the reasoning queries and render them with prompts templates to a proper language model input format.

Table 1: Prompt templates and the number of test cases (#) investigated for each numerical reasoning task.

Task	Prompt Template	#
Arithematic		
Multiplication	*Q:What is* x_1 *times* x_2*? A:* y	10^4
Addition	*Q:What is* x_1 *plus* x_2*? A:* y	10^4
Operation Inference		
Mult. #	*Q:What is* x_1 *#* x_2*? A:* y	10^4
Add. #	*Q:What is* x_1 *#* x_2*? A:* y	10^4
Time Unit Inference		
Min→Sec	*Q:What is* x_1 *minutes in seconds? A:* y	79
Hour→Min	*Q:What is* x_1 *hours in minutes? A:* y	100
Day→Hour	*Q:What is* x_1 *days in hours? A:* y	100
Week→Day	*Q:What is* x_1 *weeks in days? A:* y	100
Month→Week	*Q:What is* x_1 *months in weeks? A:* y	100
Year→Month	*Q:What is* x_1 *years in months? A:* y	100
Decade→Year	*Q:What is* x_1 *decades in years? A:* y	100

compute the average accuracy of the model over all instances where one of the operands is 23. The performance gap w.r.t. to $\omega_{\{x_1\}}$ for this task is the difference between the average accuracy over the top 10% and the bottom 10% most frequent numbers in the pretraining corpus. We picked 10% as the threshold to have a *simple*, *intuitive* metric that captures how accuracy differs between the most and least frequent terms. We also provide the plots to show the full distribution in the frequency range.

3 Experiment Setup

In this section, we describe our setup to measure the effect of pretraining data on the few-shot evaluation of a number of numerical reasoning tasks for different language models.

Language Models We experiment on the following models from EleutherAI: GPT-J-6B (Wang, 2021), and GPT-Neo-1.3B, GPT-Neo-2.7B (Black et al., 2021). These models are publicly available, but more importantly, they are among the few models that their pretraining corpus has also been released. These language models are trained on the Pile dataset (Gao et al., 2020), a large-scale language modeling dataset consisting of English documents in 22 academic or other professional data sources. We count the frequency of all integers with less than seven digits using a slightly modified version of Spacy English tokenizer (Honnibal and Montani, 2017). To calculate the frequencies of the numbers we use Amazon Elastic Map Reduce (EMR) platform. We use the HuggingFace[1] Transformer integration of the models for experiments.

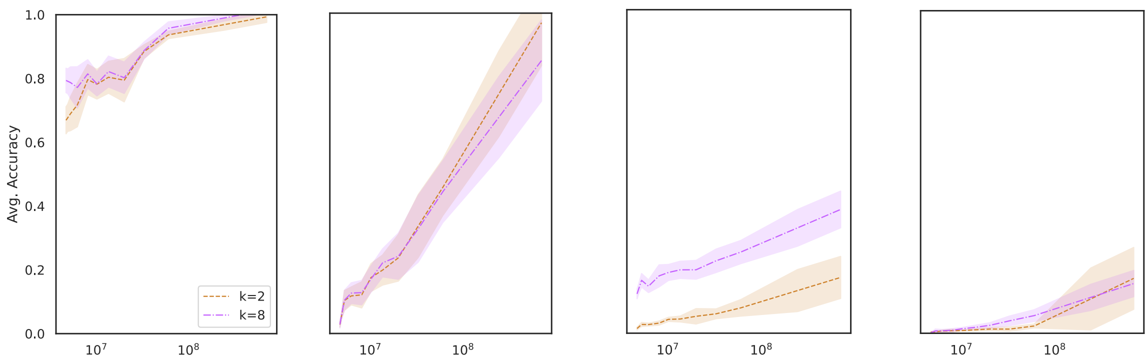
Numerical Reasoning Tasks We create three types of datasets that target the mathematical capabilities of language models since solving mathematical questions is a useful *reasoning* capability of the models (Brown et al., 2020).

- **Arithmetic, 2 tasks** As the first task, we consider simple arithmetic operations: addition $x_1 + x_2 \rightarrow y$ and multiplication $x_1 \times x_2 \rightarrow y$. In both cases, the both operands (x_1 and x_2) are numbers less than 100 (these numbers are in the top 200 most frequent numbers in the pretraining data).

- **Operation Inference, 2 tasks** Instead of directly specifying the operation, we also create a variation where the model needs to *infer*, from a few examples, the operation itself, as well as the result, as introduced in the evaluation of Megatron-Turing model.[2] We replace the arithmetic operation with a "#", with the same operations and operands as previous, to create these datasets.

- **Time Unit Conversion, 7 tasks** Apart from direct arithmetic expressions, we are also interested in evaluating model capability to implicitly reason about these operations. To this end, we construct a unit conversion dataset by identifying the most frequent numbers that co-occur with time unit words ("second", "minute", "hour", "day", "week", "month", "year", and "decade") as the primary operand x_1, the time units themselves as additional operands ($x_2 \rightarrow x_3$), i.e. converting 24

[1] Source code at `https://huggingface.co/EleutherAI`
[2] `https://turing.microsoft.com/`



(a) Arithmetic-Addition (b) Arithmetic-Multiplication (c) Op.Inference-Addition (d) Op. Inference-Mult.

Figure 3: **The GPT-J-6B accuracy on arithmetic and operator inference**, with k shots. The average accuracy of the binned instances is highly correlated with their term frequencies $\omega_{\{x_1\}}$ in the pretraining corpus (x-axis).

hours to minutes is represented as (24, “hours”, 60). We expect converting time values to be mathematically more straightforward than two-digit multiplication since the model need only multiply with the same (implicit) second operand, e.g., $\times 60$ for converting hours to minutes.

The pipeline for creating instances in our evaluation is illustrated in Figure 2. We compute occurrences and co-occurrences (assuming a window of 5) of the terms in the corpus, i.e. the time units and numbers. We generate instances for the reasoning tasks using the most frequent terms with less than 3 digits (the top 200) as operands. We focus on the top terms since we expect the models to have a fairly reliable and robust representations for these words. Each reasoning instance is *rendered* as a natural language query using the prompt templates from Table 1, and input to the language model to generate the answer. For example, to create a multiplication instance given the terms ($x_1 = 23, x_2 = 18$), we use the instance template to create a natural language input for the model as *“Q: What is* 23 *times* 18*? A:* __”, with the goal of producing “414” ($y = 23 \times 18 = 414$). For few-shot evaluation, we prompt the language models with $k = 0, 2, 4, 8, 16$ shots, and average performance over five random selection of the prompt instances.

4 Results

With the three types of numerical reasoning tasks (consisting of 11 total datasets), we present an evaluation of the effect of pretraining term frequency on the performance of the language models. For each dataset, we measure the performance gap on instances that consist of rarer (relatively) terms, for a few different choices of what to compute frequency over (different combinations of the instance terms). We also investigate the effect of the model size on this performance gap and do a case study to further clarify if all this impact is due to memorization.

Arithmetic We first study the performance on simple addition and multiplication of numbers. The results for the GPT-J-6B model is provided in Table 2, with performance gap computed just for x_1 (any of the multiplicands), for (x_1, x_2) (both multiplicands), and for (x_1, y) (any of the multiplicands and the golden answer). In *multiplication*, we observe a very high performance gap for all these definitions of frequencies, suggesting a strong effect of frequency in the pretraining data on the model’s ability to perform multiplication. For better illustration of the performance gap, we plot the mean accuracy across the frequency of x_1 in Figure 3b. The plot demonstrates the strong correlation between the models accuracy on specific instances, and the instance element frequency in the pretraining data. For *addition*, we observe an overall higher performance of the GPT-J-6B model in comparison to the multiplication experiments. However, the performance gap on all of the definitions of the instance frequencies still shows an strong effect on the models accuracy. As shown in Figure 3a, the average accuracy of the model still has a positive slope, indicating the effect of instance frequencies.

Operation Inference These tasks aim to assess the model capability to both infer the math operation and to perform the actual computation. As we see in Table 2, the model is much less accurate here as compared to the arithmetic experiments. However, the model has better performance on the frequent instances even for these low performance tasks (see detailed trend in Figures 3d and 3c). The

Table 2: **GPT-J-6B results on arithmetic, operation inference (#) tasks** Δ_1, $\Delta_{1,2}$ and $\Delta_{1,y}$ represent the performance gap over the frequency distributions of $\omega_{\{x_1\}}$, $\omega_{\{x_1,x_2\}}$ and $\omega_{\{x_1,y\}}$ respectively. x_1 represent the first operand, x_2 second operand and y the answer of the arithmetic question.

k	Multiplication				Addition				Multiplication (#)				Addition (#)			
	Acc.	Δ_1	$\Delta_{1,2}$	$\Delta_{1,y}$	Acc.	Δ_1	$\Delta_{1,2}$	$\Delta_{1,y}$	Acc.	Δ_1	$\Delta_{1,2}$	$\Delta_{1,y}$	Acc.	Δ_1	$\Delta_{1,2}$	$\Delta_{1,y}$
0	2.6	13.2	13.3	21.4	1.6	8.2	7.3	9.9	-	-	-	-	-	-	-	-
2	25.5	69.2	83.9	87.0	80.6	29.2	32.1	44.5	2.0	10.2	11.6	12.1	5.2	11.3	17.8	17.1
4	23.9	61.1	76.1	77.2	86.2	20.8	26.4	31.2	2.2	9.1	12.9	14.3	21.8	34.7	51.0	49.4
8	25.0	61.8	75.1	76.3	84.1	19.8	26.6	25.8	2.9	10.7	17.6	16.1	20.3	18.8	32.1	26.8
16	25.4	64.1	76.1	77.8	83.9	17.9	24.8	22.4	5.2	21.1	28.7	28.5	15.1	17.2	28.1	23.8

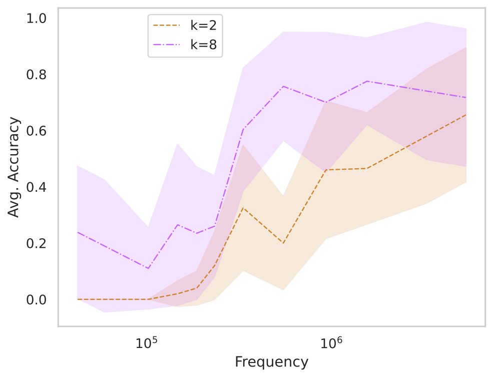


Figure 4: **GPT-J-6B performance on Year→Month:** Interpolation lines show the correlation between the average accuracy and the $\omega_{\{x_1,x_2\}}$ (k is number of shots).

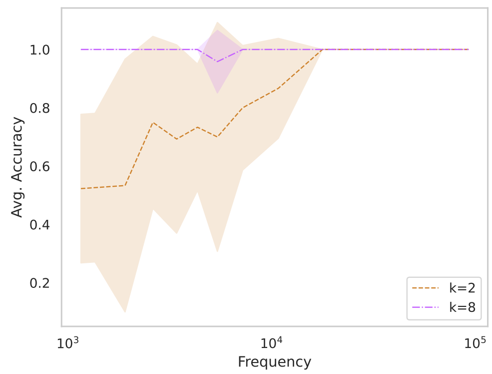


Figure 5: **GPT-J-6B performance on Decade→Year:** The interpolation average accuracy line over the $\omega_{\{x_1,x_2\}}$ show that the model reaches a high performance with the number of shots $k = 8$, there is still a performance gap in the case of $k = 2$.

performance gap here suggests that the effect of pretraining is not only for tasks that the model is accurate on, but even for operation inference that is more challenging and require deeper reasoning. Moreover, the lower accuracy here as compared to addition experiments in the previous section suggests that the model is unable to infer the operation from the few-shot prompts, and it may be performing some form of pattern matching based on the pretraining data on the common instances.

Time-Unit Conversion The performance gap evaluated on the time unit conversion experiments is in Table 3. We first observe a relatively high performance gap on all the tasks except the conversion from decade to year. We also observe a general pattern of increase in the performance gap as the number of shots (training examples in the prompt) increases. These results suggest that even though the model gets more accurate, the improvements focus on more frequent instances of the task. (Example figures for time units experiments is provided in Figures 4, 5)

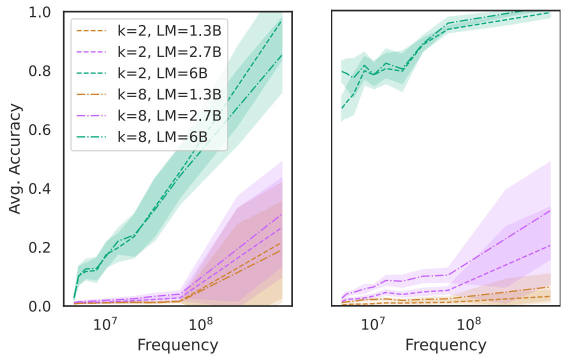
Decades to years: As we observe in Table 3, the model performs nearly perfectly on this task with as few as 8 shots, and we only see very small performance gap. This is likely due to the task being quite simple (appending a "0" to the input number) so, the model is able to *generalize* in the manner we are evaluating it. However, it is also possible that we are simply not identifying the right frequency statistics for this task, and there is an effect that our current evaluation setup does not capture.

Studying the Size of Language Models To further study the impact of language models sizes on the performance gap caused by the instance frequencies, we perform the arithmetic experiments for 2, 8 shots using the smaller models (GPT-Neo-1.3B and GPT-Neo-2.7B). We can see the trends of the average accuracy of the models in Figures 6. The smaller models overall are less accurate on the arithmetic tasks, which is consistent with observations in related work (Brown et al., 2020). However, their success is still focused on the more frequent terms from the pretraining corpus, suggesting that even the smaller models show the effect of reliance on the pretraining data, although to a much lower extent than the larger ones.

Table 3: **GPT-J-6B results on Time-Unit Conversion:** $\Delta_{1,2}$, $\Delta_{1,2,3}$ and $\Delta_{1,2,y}$ represent the performance gap over the frequency distributions of $\omega_{\{x_1,x_2\}}$, $\omega_{\{x_1,x_2,x_3\}}$ and $\omega_{\{x_1,x_2,y\}}$ respectively, where x_1 is the number operand, x_2 is the source unit, x_3 is the number operand needed for performing the conversion and the y is the true answer.

k	Min→Sec				Hour→Min				Day→Hour				Week→Day			
	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$
0	1.3	0.0	0.0	12.5	1.0	0.0	0.0	5.0	1.0	0.0	0.0	10.0	1.0	0.0	0.0	10.0
2	25.5	60.0	62.5	62.5	19.4	62.0	40.5	60.5	12.1	26.0	26.0	16.0	13.1	44.0	46.0	52.0
4	35.5	60.7	65.0	50.6	29.1	71.5	49.9	56.5	22.7	54.0	52.0	39.5	19.2	50.0	48.0	50.0
8	49.9	72.7	82.3	42.1	36.3	78.0	52.5	52.9	31.0	67.0	61.0	61.0	28.6	68.0	64.0	53.5
16	58.4	87.5	88.5	64.2	42.8	79.0	50.4	57.8	43.3	67.7	57.7	48.0	28.0	30.9	24.5	51.3

Shots, k	Month→Week				Year→Month				Decade→Year			
	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$	Acc.	$\Delta_{1,2}$	$\Delta_{1,2,3}$	$\Delta_{1,2,y}$
0	1.0	0.0	0.0	10.0	1.0	0.0	0.0	10.0	3.1	14.3	0.0	28.6
2	30.1	9.0	13.0	13.0	21.8	58.0	64.0	62.0	76.5	47.4	30.0	20.0
4	63.3	21.5	25.5	5.5	31.9	64.8	62.8	70.0	96.7	2.9	0.0	2.9
8	80.9	37.5	24.0	5.0	45.4	55.0	62.0	59.0	99.6	0.0	0.0	0.0
16	84.5	54.0	51.0	21.2	56.7	58.7	57.3	66.8	100.0	0.0	0.0	0.0



(a) Arithmetic-Mult. (b) Arithmetic-Add.

Figure 6: **The effect of model size on performance** Smaller models only perform well on instances with more frequent terms in the pretraining data. k represents the number of shots.

Impact Due to Memorization In this section, we will study the extent to which the impact of the pretraining term frequencies on model performance can be explained due to pure memorization. To tease apart direct memorization, we perform similar analysis as above, but do so without the instances that have already been memorized by the language model in their entirety. It is worth mentioning that finding such instances is not trivial. In other words, it is not trivial to identify purely memorized instances. Prior work (Magar and Schwartz, 2022; Carlini et al., 2022) has shown that the models perform more accurately on the instances with higher *exact match* counts from the pertaining data; we first verify this trend by observing a similar trend for our setting by plotting $\omega_{\{x_1,x_2,y\}}$, the co-occurrence of *all three numbers* in Appendix Figure 9). Based on these studies and results, we treat the number triples with the highest average accuracy (more than 85%), as the ones the model has most likely memorized. Specifically, we remove these memorized instances from our evaluation to see the impact of lower order term frequencies on the remaining instances. As shown in Figure 7, the dependency of model performance on lower order frequencies ($\omega_{\{x_1\}}$ and $\omega_{\{x_1,x_2\}}$) is still very high even after removing the memorized instances. These observations suggest that the impact of term frequencies on model performance is beyond pure memorization of the numerical terms.

Summary Overall, we observe high positive performance gap for almost all of the experiments on the three definition levels of the frequency for each task. This suggests a strong effect of frequency of the instances in the pretraining data on the model performance. In particular, evaluation using performance gap with $\omega_{\{x_1\}}$ shows that even the *unigram* statistics of the instances have strong correlation with the models performance on the instance.

Other than some exceptional cases, we observe an increasing trend in the performance gap as we put more training examples in the prompt (the number of shots); this can be a further indication that the model is directed through the patterns in the pretraining data to answer the reasoning questions. Our experiments with the smaller sizes of the model also show that they can only solve the frequent instances of the tasks, which further supports our observation that model performance is correlated with the term frequencies.

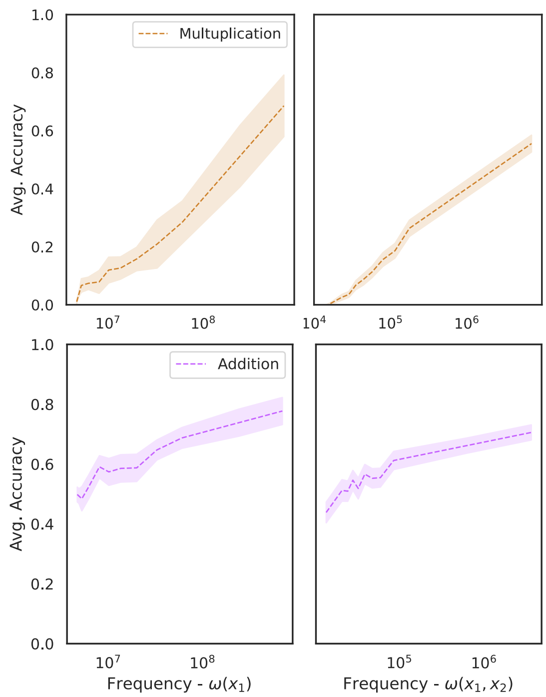


Figure 7: **The effect of term frequencies after removing memorized instances** on 2-shot GPT-J-6B. Dependence of model performance on unigram and co-occurrence frequencies (after removing the memorized instances) shows the effect exists beyond memorization.

5 Related Work

A large and growing body of literature has investigated a number of related concerns with large language models (for discussion of more tangentially related work see Appendix A.1).

Numeracy and Temporal Reasoning in LMs Our work contributes to the larger body of work studying numeracy in word embeddings and language models (Spithourakis and Riedel, 2018; Wallace et al., 2019). Geva et al. (2020), Zhou et al. (2020) Zhou et al. (2022) and Lewkowycz et al. (2022) propose training schemes to help improve LMs' temporal and numerical reasoning capabilities. Patel et al. (2021) show that NLP math solvers rely on simple heuristics to answer math questions. We expect that the performance gap metric proposed in this work will be useful to better understand the impact of such schemes.

Impact of Frequency on LM Performance Kassner et al. (2020) and Wei et al. (2021) perform controlled experiments varying pretraining data to characterize the extent to which pretraining affects LMs' ability to memorize and reason with facts as well as learn generalizable syntax rules. In line with our results, both of these find that frequency is a distinguishing factor in whether or not the model memorizes a particular fact or syntactic rule for a verb form. Sinha et al. (2021) further demonstrate that shuffling word order during pretraining has minimal impact on an LMs' accuracy on downstream tasks, and, concurrent with this work, Min et al. (2022) similarly find that shuffling labels in in-context learning demonstrations has minimal impact on few-shot accuracy. These results further suggest that LMs' performance is largely driven by their ability to model high-order word co-occurrence statistics. Data privacy researchers have shown that LMs may memorize sensitive sequences occurring in training data even if they are rare (Carlini et al., 2019; Song and Shmatikov, 2019).

Memorization Feldman (2020) provide a theoretical definition of memorization as the difference between the accuracy of a model on a training data point when that point is included vs. excluded from training. They also develop an approach for approximating memorization using influence functions Feldman and Zhang (2020). This framework is applied to study memorization in language models by Zhang et al. (2021), who find that training examples that are memorized by the LM tend to have high influence of LM predictions on similar validation instances. Their result may provide a plausible explanation that the frequency effects observed in this work are due to memorization.

6 Discussion

In this work, we consider how to conduct few-shot evaluations in light of the analysis with the pretraining data. Prior work has attempted to control for overlap between pretraining data and the test instances, but as we have seen, those methods are insufficient. For example, Brown et al. (2020) measure the impact of removing instances from evaluation datasets that share 13-gram overlap with their pretraining data on GPT-3's accuracy, and also argue that the low occurrence of exact phrases such as "NUM1 + NUM2 =" and "NUM1 plus NUM2" in the pretraining data indicate that the model's strong performance on arithmetic tasks is likely due to factors other than memorization. However, we show that LM performance is impacted by much simpler statistical patterns, as small as unigram overlap with the pretraining data.

For these reasons, we strongly recommend that evaluation of reasoning capabilities should take the pretraining corpus into account, and any claims of reasoning can only be made after demonstrating robustness to the effect of pretraining. Current LM benchmarks, that are dissociated from the model's pretraining data, make it impossible to interpret few-shot reasoning performance results. It is worth mentioning that, even a performance gap of 0 is likely not sufficient to claim reasoning capabilities—what exactly constitutes "reasoning" remains ill-defined—but it may be a necessary condition, and one that current models do not meet.

In this study, we are not making a *causal* claim, and in general, there may be confounders that we have not eliminated in our setting. Recently, Elazar et al. (2022) introduced a causal framework based on pretraining data statistics for understanding language model's *factual predictions*. To be able to use the causal inference techniques they construct and assume a causal graph for the task of extracting factual knowledge from pretrained language models. We recommend further research in the proposed direction for other NLP tasks such as reasoning and interventions during training to provide finer-grained analysis of the effect of pretraining.

One potential concern is that our experiments do not distinguish whether incorrect answers are due to lack of reasoning or lack of recognition, i.e. it is possible that the model has the ability to multiply but the embeddings for rare terms are not adapted to that algorithm. However, recognizing numbers is a prerequisite to numerical reasoning, thus if the models lack the ability to identify numbers, this still means that they lack numerical reasoning skills. That said, we also suspect that the errors are not due to recognition. Even the most infrequent terms in our experiments have been seen *millions* of times—they are not unknown tokens.

7 Conclusion

We show that in-context language model performance on numerical reasoning tasks can be impacted significantly by low-order co-occurrence statistics in the pretraining data, raising questions on the extent to which these models are actually *reasoning* to solve these tasks. These observations suggest the necessity for reconsidering and redefining the reasoning evaluation schemes for the large language models. Further characterizing the impacting factors on the models reasoning capacities is also an important tasks for the community. Most importantly, we suggest that the community should not treat the pretraining data of the large language models as unknown black boxes. Overlooking the impact of the pretraining data can be misleading in evaluating the model reasoning skills.

Acknowledgements

We would like to thank the members of UCI-NLP, Yanai Elazar, Mukund Sundarajan, Marco Tulio Ribeiro, Eric Wallace, Shivanshu Gupta, Navid Salehnamadi, Pouya Pezeshkpour, and Dylan Slack for valuable discussions and feedback on this work. This material is sponsored in part by the DARPA MCS program under Contract No. N660011924033 with the United States Office Of Naval Research, by an Amazon Research Award, and by awards IIS-2046873 and IIS-2040989 from the National Science Foundation.

Limitations

There are a few limitations to our study that open up avenues for future research. First, our approach aggregates fairly simple patterns and the effect we observe might be stronger if a wider variety and complexity of patterns is considered in the pretraining corpus. Similarly, our work is limited to simple numerical reasoning tasks, and it would be worthwhile to study how much other reasoning evaluations and more complex quantitive reasoning tasks such as GSM8K (Cobbe et al., 2021) are impacted by the same effect, which could be measured using the performance gap metric introduced here. Defining appropriate instance terms for other reasoning tasks such as commonsense reasoning will be a challenging but important direction for future work. Lastly, we do not propose a solution for changing language models to robust reasoners. We hope that the insights in this work inspire further studies into the effect of pretraining on language model's performance, improvements in evaluation schemes, and better training mechanisms for more robust language models with true generalization capabilities.

References

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 267–284, USA. USENIX Association.

Alvin Chan, Yi Tay, Yew-Soon Ong, and Aston Zhang. 2020. Poison attacks against text datasets with conditional adversarially regularized autoencoder. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4175–4189, Online. Association for Computational Linguistics.

Stephanie CY Chan, Adam Santoro, Andrew K Lampinen, Jane X Wang, Aaditya Singh, Pierre H Richemond, Jay McClelland, and Felix Hill. 2022. Data distributional properties drive emergent in-context learning in transformers. *arXiv preprint arXiv:2205.05055*.

Anthony Chen, Pallavi Gudipati, Shayne Longpre, Xiao Ling, and Sameer Singh. 2021. Evaluating entity disambiguation and the role of popularity in retrieval-based NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4472–4485, Online. Association for Computational Linguistics.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, Online. Association for Computational Linguistics.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Amir Feder, Abhilasha Ravichander, Marius Mosbach, Yonatan Belinkov, Hinrich Schütze, and Yoav Goldberg. 2022. Measuring causal effects of data statistics on language model'sfactual'predictions. *arXiv preprint arXiv:2207.14251*.

Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 954–959. ACM.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III au2, and Kate Crawford. 2021. Datasheets for datasets.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8706–8716, Online. Association for Computational Linguistics.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are pretrained language models symbolic reasoners over knowledge? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 552–564, Online. Association for Computational Linguistics.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021a. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021b. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020a. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020b. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Alexandra Luccioni and Joseph Viviano. 2021. What's in the box? an analysis of undesirable content in the Common Crawl corpus. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2:*

Short Papers), pages 182–189, Online. Association for Computational Linguistics.

Inbal Magar and Roy Schwartz. 2022. Data contamination: From memorization to exploitation. *arXiv preprint arXiv:2203.08242*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Blaine Nelson, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I. P. Rubinstein, Udam Saini, Charles Sutton, J. D. Tygar, and Kai Xia. 2008. Exploiting machine learning to subvert your spam filter. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, USA. USENIX Association.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Yasaman Razeghi, Raja Sekhar Reddy Mekala, Robert L. Logan IV, Matt Gardner, and Sameer Singh. 2022. Snoopy: An online interface for exploring the effect of pretraining term frequencies on few-shot lm performance. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Alexey Romanov, Maria De-Arteaga, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, Anna Rumshisky, and Adam Kalai. 2019. What's in a name? Reducing bias in bios without access to protected attributes. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4187–4195, Minneapolis, Minnesota. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 8–14, New Orleans, Louisiana. Association for Computational Linguistics.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Serge Sharoff. 2020. Know thy corpus! robust methods for digital curation of web corpora. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2453–2460, Marseille, France. European Language Resources Association.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2888–2913, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Congzheng Song and Vitaly Shmatikov. 2019. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 196–206. ACM.

Georgios Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2104–2115, Melbourne, Australia. Association for Computational Linguistics.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What makes reading comprehension questions easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Brussels, Belgium. Association for Computational Linguistics.

Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing numbers in NLP: a survey and a vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–656, Online. Association for Computational Linguistics.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Eric Wallace, Tony Zhao, Shi Feng, and Sameer Singh. 2021. Concealed data poisoning attacks on NLP models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 139–150, Online. Association for Computational Linguistics.

Ben Wang. 2021. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. `https://github.com/kingoflolz/mesh-transformer-jax`.

Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. 2021. Frequency effects on syntactic rule learning in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 932–948, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. Probing neural language models for human tacit assumptions. In *CogSci*.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2021. Counterfactual memorization in neural language models. *arXiv preprint arXiv:2112.12938*.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal common sense acquisition with minimal supervision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7579–7589, Online. Association for Computational Linguistics.

Fan Zhou, Haoyu Dong, Qian Liu, Zhoujun Cheng, Shi Han, and Dongmei Zhang. 2022. Reflection of thought: Inversely eliciting numerical reasoning in language models via solving linear systems. *arXiv preprint arXiv:2210.05075*.

A Appendix

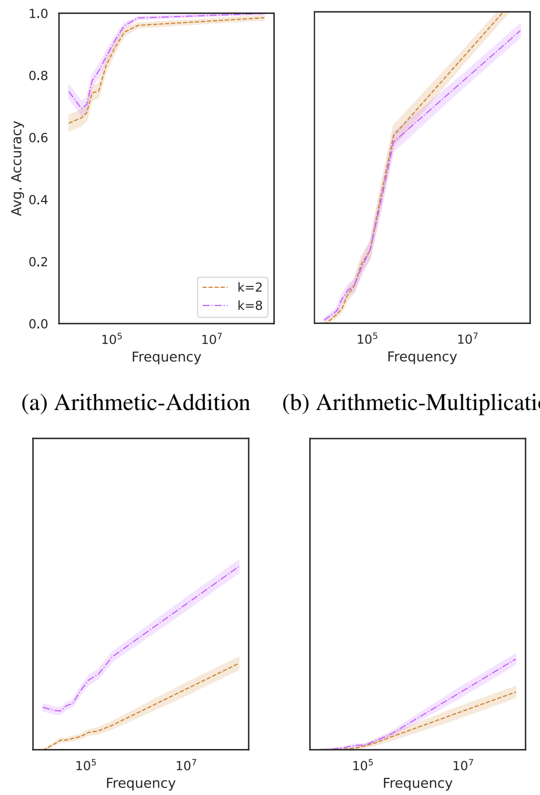
A.1 Additional Related Work

In this section, we further discuss the related work.

Prompting Prompting has been widely applied to study the factual (Petroni et al., 2019), commonsense (Davison et al., 2019; Weir et al., 2020; Lin et al., 2020b), mathematical (Saxton et al., 2019), and other NLP task-related (Radford et al., 2019; Shin et al., 2020) knowledge LMs acquire during pretraining. In this work, we focus on the *in-context learning* setup of Brown et al. (2020), who use prompts that include training examples to diagnose LMs' few-shot learning capabilities.

Training Artifacts Challenge Evaluation Our results raise the issue that in-context learning probes may overestimate an LM's ability generalize from few examples when biases are present in the training data. This is consistent with prior work that has exposed the similar effects of biases from: lexical cues in natural language inference datasets (Gururangan et al., 2018; Poliak et al., 2018; McCoy et al., 2019), question-passage overlap and entity cues in reading comprehension datasets (Chen et al., 2016; Sugawara et al., 2018; Jia and Liang, 2017; Lewis et al., 2021b), gender cues in coreference resolution datasets (Rudinger et al., 2018), popularity in named entity disambiguation (Chen et al., 2021), similarity between training and test instances in information extraction and sentiment analysis datasets (Elangovan et al., 2021), and effects of how data is split (Gorman and Bedrick, 2019; Søgaard et al., 2021). Relatedly, data poisoning research studies how to adversarially introduce artifacts into training data to produce unwanted model behaviors (Nelson et al., 2008; Chan et al., 2020; Wallace et al., 2021). A general statistical procedure to test for artifacts is presented in Gardner et al. (2021), who also theoretically show that large datasets are almost certain to contain artifacts under reasonable assumptions. Techniques for mitigating biases in the presence of dataset artifacts are covered by Romanov et al. (2019) and Karimi Mahabadi et al. (2020).

Documenting Pretraining Data To better understand the risks of dataset artifacts, there has been a call to better document the characteristics and intended uses of datasets (Gebru et al., 2021; Bender et al., 2021). However, due to the sheer size of LM pretraining datasets—which range from 100's of GBs to 10's of TBs—doing so can pose a substantial challenge. Despite this, researchers have been able to estimate word frequencies, topics, and genres of documents (Sharoff, 2020), as well as proportions of toxic text (Gehman et al., 2020) appearing in OpenWebText (Gokaslan and Cohen, 2019). Similar efforts have been made to characterize the top-level domains, amount of hate speech, and censured text appearing in the C4 corpus (Raffel et al., 2020; Dodge et al., 2021; Luccioni and Viviano, 2021). Our work documents co-occurrence statistics of numbers and dates of documents appearing in the Pile dataset.



(a) Arithmetic-Addition (b) Arithmetic-Multiplication

(c) Op.Inference-Addition (d) Op. Inference-Mult.

Figure 8: **The GPT-J-6B accuracy on arithmetic and operator inference** tasks, with k shots. The average accuracy (y-axis) of the binned instances is highly correlated with their co-occurrences term frequencies $\omega_{\{x_1,x_2\}}$ in the pretraining corpus (x-axis).

A.2 Examples of time unit conversion plots

We provide the figures showing the dependence between the average accuracy and the $\omega_{\{x_1,x_2\}}$ for time unit experiments of Minute, Year and Decade in Figures 10, 4 and 5, respectively.

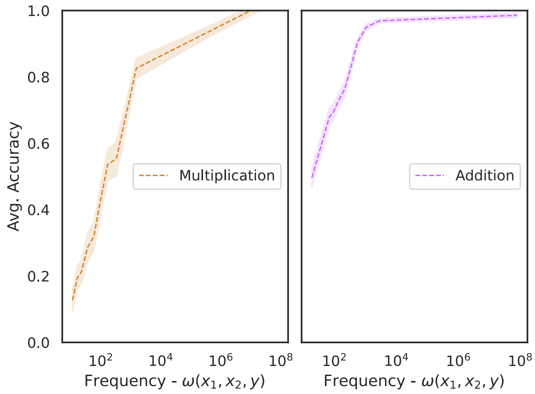


Figure 9: The impact of $\omega_{\{x_1,x_2,y\}}$ (the frequency of all numbers (x_1, x_2, y) in an arithmetic instance) on GPT-J-6B's 2-shot performance, the high dependence of models average accuracy on $\omega_{\{x_1,x_2,y\}}$ may be due to memorization specifically in highest frequencies

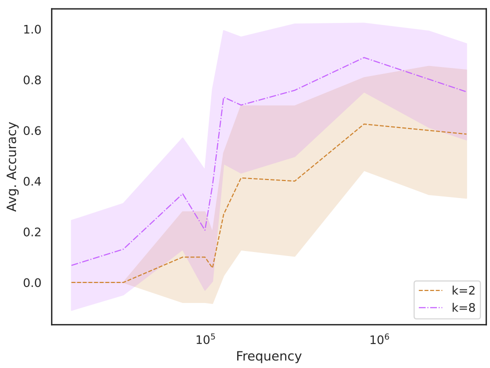


Figure 10: **GPT-J-6B performance on Minute→Second:** The interpolation lines show the correlation between the average accuracy and the $\omega_{\{x_1,x_2\}}$. k is the number of shots.