Contamination Budget: Trade-offs Between Breadth, Depth and Difficulty

Behzad Mehrbakhsh[1,2], **Fernando Martínez-Plumed**[1,2], **José Hernández-Orallo**[1,2]
[1]UPV - Universitat Politècnica de València
[2]VRAIN - Valencian Research Institute for Artificial Intelligence
bmehrba@upv.es, fmartinez@dsic.upv.es, jorallo@upv.es

Abstract

Contamination in large language models (LLMs), and machine learning more broadly, refers to the inclusion of equal –or very similar– examples in both training and test sets. This phenomenon usually translates into better test performance. Here we explore when this contamination is performed intentionally, for purposes that can be malicious (e.g., get better scores in evaluations) or benevolent (e.g., fix some mistakes). These interventions, usually in the form of fine-tuning memorisations, come with a budget in the size of the fine-tuning dataset. Several trade-offs appear between the breadth of the intervention (how many examples to be memorised), its depth (how many repetitions of each example) and the difficulty of the examples. By studying several LLMs and datasets, we observe some monotonic behaviour (more difficult items require more depth to be 'fixed') but also some non-monotonic phenomena (very high depth levels have negative effects on non-contaminated examples). This suggests that trade-offs should be found not only in terms of the budget but also according to model specifics, the task and the item difficulty at hand.

1 Introduction

Evaluation contamination is regarded as one of the major problems for the rigorous assessment of LLMs [Chang *et al.*, 2024; Zhou *et al.*, 2023]. This is so because, contrary to other machine learning models, the size and diversity of the data used during the training of LLMs makes it quite challenging to ensure that there is no intersection between the training and test sets. Also, a great part of the desired behaviour depends on the memorisation of facts and knowledge. For instance, we cannot expect a model to answer the question "`What's the capital of France?`" if the model has not been trained or fine-tuned with that information. This makes contamination an ill-defined phenomenon affected by a tension between memorisation and generalisation [Magar and Schwartz, 2022; Dong *et al.*, 2024].

Contamination is therefore seen as a fatality that occurs unintentionally and unavoidably, and the effort should be put on detecting it and palliating its effects [Deng *et al.*, 2024]. It

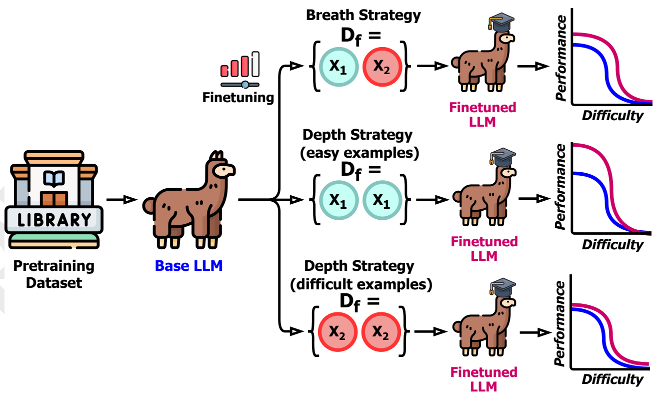

Figure 1: Contamination intervention strategies under a fixed fine-tuning budget $b = 2$. We examine the trade-offs between breadth (number of varied examples), depth (repetitions of each example), and example difficulty (difficult in **red**, easy ones in aquamarine).

is poorly recognised though that contamination can be introduced on purpose. First, a malicious or interested actor (e.g., an AI provider or researcher) can contaminate the training set with many examples that are expected in the test set, inflating the reported performance of the model. Second, a benevolent actor (e.g., an AI provider or researcher) may be interested in fixing some very common mistakes by introducing them in the training set. Independently of the malicious or benevolent motivation, both interventions are highly similar and are affected by the same bottleneck: a model cannot memorise all the examples that may be expected during deployment. It may be possible for the small list of capitals of countries, but not for vast encyclopaedic knowledge, or tasks that have an infinite number of instances such as addition. In practice, we have a *budget* of data points that we can introduce during training (either pre-training or fine-tuning). Given this budget, we would like to maximise the effect of the intervention.

Intentional contamination should focus on fine-tuning with those examples where the model failed after pre-training but are most frequent during deployment (and testing). In this context, we came up with two observations. First, if a model fails on two questions x_1 = `What's the capital of France?` and x_2 = `What's the capital of Bhutan?`, and the budget limits the 'breadth' of contamination (how many examples we contaminate), it seems more utilitarian to fix x_1 first rather than x_2, as the expected frequency is higher, and hence expected performance will be more affected, assuming that both are equally easy to fix. Second, and mostly overlooked by the literature of memorisation [Carlini *et al.*, 2019] and contamination [Magar and Schwartz, 2022], it makes more practical sense to fix the easiest examples first. Also, it seems that easy examples are typically more frequent. We can now translate these observations into fine-tuning interventions.

Given these observations, we now face a fundamental question: is it better to prioritise for breadth (trying to fix more examples), for depth (doing more repetitions of a few examples) or for difficulty (selecting those easy examples first)? For instance, Figure 1 shows the case with a budget of $b = 2$ examples. In this situation, would it be more effective to build a fine-tuning dataset $D_\mathsf{f} = \{x_1, x_2\}$, $D'_\mathsf{f} = \{x_1, x_1\}$, or $D''_\mathsf{f} = \{x_2, x_2\}$? Perhaps D_f, which prioritises breadth, does not have enough repetitions of each example (depth) to fix any of the two, whereas D'_f could at least succeed in fixing x_1? There seems to be a trade-off. However, is it clear that D''_f is the worst option of the three, assuming that x_2 is a more difficult and less frequent example?

Both malicious and benevolent actors have strong motivations to operate following the above rationale. If a malicious actor wants to contaminate a model to score better in the test sets of a benchmark, they will try to fix the common and easy-to-fix instances first, in order not to raise suspicion. If a benevolent actor wants to contaminate a model to fix instances to increase reliability they should operate on those that are common and easy-to-fix, especially because reliability is expected in the easy instances [Zhou *et al.*, 2024].

In this paper, we formulate new and fundamental research questions about the most effective use of a budget on the number of examples for fine-tuning a pre-trained LLM. We answer them by exploring several models in different families, and four datasets (MMLU-Pro [Wang *et al.*, 2024], MedMCQA [Pal *et al.*, 2022], addition and anagram [Zhou *et al.*, 2024]) that are representative of either multiple-choice and open-ended questions, more or less generalisable domains (addition vs MedMCQA), etc. We analyse the effect with different budgets and strategies on the contaminated examples and the rest, in either in-distribution and out-of-distribution scenarios. Our main finding is that excessive depth can have negative effects in-distribution but this does not translate out-of-distribution. In the end depth, should be prioritised, and the trade-off between breadth and depth should be chosen according to some monotonocities and non-monotonicities seen in the effectiveness of the contamination intervention.

2 Budget, Breadth, Depth and Difficulty

Let $x_i \in \mathcal{X}$ denote the i-th instance (or example), from instance space $\mathcal{X}$, and let m be a LLM, understood as a function $m : \mathcal{X} \to \mathcal{Y}$, with $\mathcal{Y}$ being the output space. We can score the outputs with a scoring or validity function $v : \mathcal{X}, \mathcal{Y} \to \mathbb{R}$ (e.g., being 1 if correct and 0 if incorrect). Given a task instance distribution $p(x_i)$ we want to maximise the following:

$$\sum_{x \in \mathcal{X}} p(x_i) \cdot v(x_i, m(x_i)) \qquad (1)$$

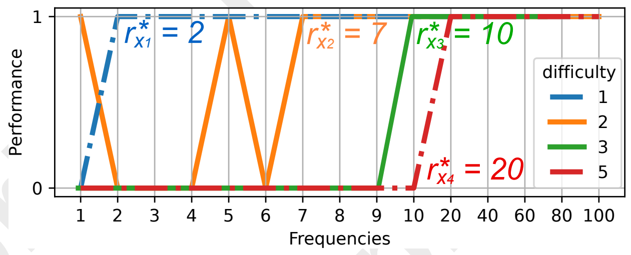

Figure 2: Illustration of f_i^m curves for the fixing of different examples (with varying difficulty) from MMLU-Pro as a function of the frequency (number repetitions r) used to fine-tune `Llama2-7B`.

Making the model produce outputs with high validity is a good strategy, but it is even better if resources are used to improve validity in the most frequent instances given $p(x_i)$.

Apart from frequency, instances can be distinguished by their difficulty or hardness $h(x_i)$ [Martínez-Plumed *et al.*, 2022; Mehrbakhsh *et al.*, 2023]. The function h can be estimated from an intrinsic or extrinsic proxy (e.g., the number of digits in an addition problem, or the proportion of a population of models or humans that fail on that instance).

Imagine a model m is wrong on instances $X_w \subset \mathcal{X}$, with $n = |X_w|$. We can try to fix m by fine-tuning on X_w, preferably multiple times, given that fine-tuning is more effective with repetitions [Carlini *et al.*, 2019]. This would require a fine-tuning dataset $D_\mathsf{f} = D^r_* = X^r_1 \uplus X^r_2 \uplus ... \uplus X^r_n$ where $\uplus$ represents the union between multisets, and each X^r_i is a multiset with r repetitions of each example $x_i \in X_w$. In total, the cardinality of $|D_\mathsf{f}| = r \cdot n$ may be quite large or even infinite. In practical scenarios, however, we only have a budget of examples b for D_f in such a way that $|D_\mathsf{f}| \leq b$. Under this situation, we have to make a selection $D_\mathsf{f} \subsetneq D^r_*$.

This selection can be performed as a function of $p(x_i)$, the budget and some other parameters. For instance, we could decide to repeat each example r times, and assuming the examples $x_1, x_2, ..., x_n$ are sorted by $p(x_i)$ in decreasing order, the selection strategy σ would be:

$$\sigma(\mathcal{X}, b, r, p) = X^r_1 \uplus X^r_2 \uplus ... \uplus X^r_{b/r} \qquad (2)$$

Let us define $f_i^m(r)$ as a function mapping the number of repetitions used for fixing instance x_i to observed validity (see Figure 2). If we apply fine-tuning to a wrong instance, then f_i^m will start at $r = 0$ until (perhaps tentatively, as for x_2 in the figure), r^*_i is reached, the smallest number of repetitions such that a sure fix happens for all $r \geq r^*_i$. This value r^*_i can be seen as *the difficulty for fixing* x_i. Of course this function f_i^m is a simplification of the actual function relating the model, repetitions and fixes. However, if r^* is monotonic and equal for every example ($\forall i, j : r^* = r^*_i = r^*_j$), then the strategy in Eq. 2 maximises Eq. 1 given the budget b (assuming validity is in $\{0, 1\}$). See the proof in Appendix B [Mehrbakhsh *et al.*, 2025].

However, not all examples require the same number of repetitions to be fixed, as seen in Figure 2. Although many factors may be involved, (1) the difficulty of the examples $h(x_i)$ may be correlated with the difficulty to fix them (r^*_i). Furthermore, (2) the difficulty of the examples $h(x_i)$ may be inversely correlated with the probability of the examples $p(x_i)$.

3 Research Questions

We aim to systematically investigate how intentional contamination interventions can be optimally designed within a limited fine-tuning budget. Specifically, we seek to understand the interplay between the breadth of the intervention (number of unique examples), the depth of the intervention (number of iterations per example), and the difficulty of the examples in effectively improving model performance. To this end, we formulate the following research questions:

RQ1: What is the **optimal trade-off between breadth and depth** of contamination interventions under a fixed fine-tuning budget to maximise model performance?

RQ2: How does the **example difficulty influence the number of repetitions** needed for effective fixation when performing contamination interventions?

RQ3: Are there negative effects or **diminishing returns associated with high depths** ($r_i \gg r_i^*$) of contamination interventions?

RQ4: How do contamination interventions affect **model performance in out-of-distribution scenarios?**

RQ5: Based on ablation studies, how do model **architecture and size affect** the success of breadth and depth strategies in LLM contamination interventions?

4 Methods

4.1 Tasks

We select recently published benchmarks to minimise the likelihood of pre-training contamination, and multiple-choice and open-ended tasks whose responses can be validated by exact string matching. MMLU-Pro [Wang *et al.*, 2024], an enhanced version of the MMLU benchmark [Hendrycks *et al.*, 2020], including more challenging, reasoning-oriented questions, increasing the number of answer choices from four to ten, and removing trivial and noisy questions found in MMLU. MedMCQA [Pal *et al.*, 2022], a dataset of medical multiple-choice questions designed to assess clinical knowledge comprehension (e.g., real-world medical entrance exam questions). Addition [Zhou *et al.*, 2024], containing 10,000 addition instances with addends ranging from 3 to 9 digits. Anagram [Zhou *et al.*, 2024], a set of 9,000 anagrams of common English words from the Google Web Trillion Word Corpus with a length ranging from 3 to 7.

4.2 Difficulty estimation

For each benchmark, we evaluate the difficulty of each test instance ($x_i \in D_\mathsf{t}$) using $K = 11$ different LLMs, denoted as $\{m_1, m_2, \dots, m_K\}$, listed in Table 1. We then compute the average correctness $\bar{v}(x_i)$ across all models:

$$\bar{v}(x_i) = \frac{1}{K}\sum_{k=1}^{K} v(x_i, m_k(x_i))$$

The estimated difficulty $h(x_i)$ of instance x_i is defined as $h(x_i) = 1 - \bar{v}(x_i)$. We divide the instances into five bins representing quintiles from the easiest to the hardest instances.

We have strong intuitions about $p(x_i)$ in the context of certain datasets, such as the addition and anagram datasets. In these datasets, instances involving smaller numbers (in addition) or shorter words (in anagrams) are more common, implying that $p(x_i)$ decreases as the difficulty $h(x_i)$ increases. Indeed, previous work has observed correlations between instance size, probability and difficulty [Zhou *et al.*, 2024].

4.3 Training, test and validation subsets

All the datasets in §4.1 are divided into several subsets:

- **Test set**: $D_\mathsf{t} \subset \mathcal{X}$, used to evaluate the performance of the models after fine-tuning.
- **Fine-tuning set**: D_f, including a series of n instances from D_t one or more times according to different contamination scenarios (see §4.5).
- **Non-contaminated set**: $D_\mathsf{u}= D_\mathsf{t} \setminus D_\mathsf{f}$, which consists of the instances in the test set that are *not* fine-tuned.
- **Validation set**: $D_\mathsf{v} \subset \mathcal{X}$, where $D_\mathsf{v} \cap D_\mathsf{t} = \emptyset$ (so no contaminated instances), used to monitor performance after fine-tuning.

The total number of instances in D_f is constrained by the contamination budget $b = \sum_{i=1}^{n} r_i$. Under a fixed b, we construct the fine-tuning set by selecting instances from D_t with specified repetitions. D_f is thus a multiset formed by the union of repetitions of the selected instances:

$$D_\mathsf{f} = \biguplus_{i=1}^{n} \{x_i\}^{(r_i)}$$

where $\{x_i\}^{(r_i)}$ is a multiset containing x_i repeated r_i times.

We chose $|D_\mathsf{t}| = b = 500$ items[1]. For D_t we randomly sample 100 instances from each of the five difficulty bins (see §4.2). Apart from the use of difficulty, we do not condition on the examples being correct or incorrect, to avoid having different sets for each model. D_f is sampled from the test set based on the scenarios in §4.5. A similar methodology was used to construct D_v, which also consists of 100 items per difficulty bin, totalling 500 items.

4.4 Models

We use a diverse set of open-access LLMs from different families and sizes to investigate how model characteristics influence the effectiveness of contamination interventions. The (open access) models selected for fine-tuning have been published in HuggingFace[2] and are described in Table 1.

To isolate the effects of contamination, we include **baseline** experiments in which the non-fine-tuned versions of the models in Table 1 are tested on D_t.

4.5 Contamination budget

To systematically explore the effects of contamination, we have introduced a *contamination budget* b, representing the total number of test item occurrences that can be included in

[1]In preliminary experiments we explored different values for these parameters which resulted in equivalent results.

[2]https://huggingface.co/

Company	Model Name	Family	Size	Version
Meta	`Llama-3.2-1B-Instruct`*	Llama 3	1B	v3.2 instruct
	`Llama-3.2-3B-Instruct`*		3B	v3.2 instruct
	`Llama-3.1-8B-Instruct`*		8B	v3.1 Instruct
	`Llama-2-7b-chat-hf`*	Llama 2	7B	v2 chat
	`Llama-2-13b-chat-hf`*		13B	v2 chat
Microsoft	`Phi-3-mini-4k-instruct`*	Phi 3	3.8B	v3 instruct
	`Phi-3-medium-4k-instruct`*		14B	v3 instruct
Mistral AI	`Mistral-7B-Instruct-v0.2`*	Mistral	7B	v0.2 instruct
TinyLlama	`TinyLlama-1.1B-Chat-v1.0`*	TinyLlama	1.1B	v1.0 chat
Google	`gemma-2-2b-it`	Gemma 2	2B	v2 instruct
	`gemma-2-9b-it`		9B	v2 instruct

Table 1: List of LLMs used in our experimental setting for fine-tuning (*) and difficulty estimation.

	$\mathbf{s_1}$	$\mathbf{s_2}$	$\mathbf{s_3}$	$\mathbf{s_4}$	$\mathbf{s_5}$	$\mathbf{s_6}$	$\mathbf{s_7}$	$\mathbf{s_8}$	$\mathbf{s_9}$
$\mathbb{E}[\mathbf{n}]$	500	250	166.7	125	100	83.3	71.4	62.5	55.6
$\mathbb{E}[\mathbf{n_j}]$	100	50	33.3	25	20	16.7	14.3	12.5	11.1
	$\mathbf{s_{10}}$	$\mathbf{s_{20}}$	$\mathbf{s_{40}}$	$\mathbf{s_{60}}$	$\mathbf{s_{80}}$	$\mathbf{s_{100}}$	$\mathbf{s_{167}}$	$\mathbf{s_{250}}$	$\mathbf{s_{500}}$
$\mathbb{E}[\mathbf{n}]$	50	25	12.5	8.3	6.3	5	3	2	1
$\mathbb{E}[\mathbf{n_j}]$	10	5	2.5	1.7	1.3	1	0.6	0.4	0.2

Table 2: Summary of contamination scenarios, all with $|D_{\mathsf{f}}|$=500. We show the frequency per item s_r, where r denotes the repetitions, the expected number of unique fine-tuning items ($\mathbb{E}[n]$), and the expected number of items per difficulty bin ($\mathbb{E}[n_j]$).

the fine-tuning dataset (D_{f}). Within this fixed budget, we vary the number n of contaminated test items and their repetitions r_i across various **contamination scenarios**:

1. **Maximum breadth, minimum depth**: Each instance appears once ($r_i = 1$), so the number of unique instances is $n = b$. Then, $D_{\mathsf{f}} = \{x_1, x_2, \ldots, x_b\}$.
2. **Intermediate breadth and depth**: Each instance appears r times ($r_i = r$ for all i), so the number of unique instances is $n = b/r$. Then, $D_{\mathsf{f}} = \biguplus_{i=1}^{n} \{x_i\}^{(r)}$.
3. **Maximum depth, minimum breadth**: Only one instance is selected ($n = 1$) and this instance is repeated b times ($r_1 = b$). Then, $D_{\mathsf{f}} = \{x_1\}^{(b)}$.

Apart from the two extreme cases of maximum breadth and maximum depth, we design several intermediate contamination scenarios to explore different allocations of the contamination budget between breadth and depth. In each scenario, the total number of fine-tuning instances is equal to the contamination budget ($b = |D_{\mathsf{f}}| = 500$). Each scenario is labelled $s_1, s_2, \ldots, s_{500}$, where the subscript denotes the frequency (number of repetitions r_i) of each selected instance in D_{f}. We ensure a uniform distribution across difficulty levels by selecting instances equally from each difficulty bin. Table 2 provides detailed information for each scenario.

4.6 Fine-tuning

We prepare the fine-tuning datasets D_{f} using the contamination strategies from §4.5, ensuring uniform difficulty distribution, and then fine-tune the LLMs in Table 1. We use the same fine-tuning hyperparameters in all experiments: learning rate (4×10^{-4}), batch size (8), number of epochs (1), optimiser (`paged_adamw_8bit` [Loshchilov, 2017]). LoRA [Yu *et al.*, 2023] is employed for parameter-efficient fine-tuning, with low-rank adaptation ($r = 16$), a scaling factor ($\alpha = 32$), and a dropout rate of 0.05. All fine-tuning experiments are performed using the HuggingFace Transformers library[3] with PyTorch [Paszke *et al.*, 2019] as the back-end.

For the tasks in §4.1, we format inputs and outputs to guide the model in selecting the correct option from given choices or completing expected pairs (see Table 4 in the appendix [Mehrbakhsh *et al.*, 2025]). After fine-tuning, we evaluate the models on the test set D_{t}, which contains both contaminated D_{f} and non-contaminated D_{u} instances, and on the validation set D_{v}, containing additional unseen non-contaminated instances.

5 Results

We now try to resolve the above research questions.

Optimal trade-off between breadth and depth (RQ1)

Figure 5 ("All items") shows the aggregated performance for all models across all tasks. Both breadth and depth have a significant impact on model performance (for test D_{t}, fine-tuned D_{f} and validation D_{v} sets). In general, increasing depth (the number of repetitions per example) improves performance on D_{f}. However, this improvement has diminishing returns and may lead to overfitting, negatively impacting non-contaminated items (D_{u}) and overall model generalisation. Conversely, prioritising breadth by introducing more unique examples increases generalisation. This approach offers slight improvements on contaminated items (D_{f}) up to a certain point (up to s_{10}), compared to s_1, without affecting performance on D_{v} or D_{u}.

Of course, the optimal balance between breadth and depth depends on the specific characteristics of the task at hand. In the anagram dataset, increased repetitions reduce the performance on D_{u} compared to the maximum breadth scenario s_1, also obtaining performance values below the baseline for D_{v}. This demonstrates overfitting due to excessive depth. The addition dataset shows a similar pattern, also suggesting diminishing returns but with almost no results below the baseline in this case for D_{v}. For MMLU-Pro and MedMCQA, performance remains relatively constant for D_{t}, while for D_{f} it increases linearly with more repetitions, and the effect on non-contaminated D_{u} items and the validation set is less pronounced, slightly decreasing with many repetitions.

As shown in Figures 7, 8, 9 and 10, in the appendix [Mehrbakhsh *et al.*, 2025] fixing certain instances—reaching 100% accuracy—is possible, but it comes with the cost of performance loss in D_{t}, D_{u} and D_{v}.

Difficulty and fixing efficiency (RQ2)

Figure 3 presents the performance of fine-tuned models on the subset of D_{f} where the original model—prior to fine-tuning—failed to provide correct responses. In the Anagram benchmark, the results indicate that the model improvement is more pronounced for the easiest difficulty bins. This observation is supported by the negative correlation between the difficulty level of the items in each bin and the aggregated

[3]https://github.com/huggingface/transformers

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{20}	s_{40}	s_{60}	s_{80}	s_{100}
Anagram	-0.98	-0.99	-0.91	-0.97	-1.00	-0.98	-0.92	-0.92	-0.95	-0.61	-0.97	0.46	-0.43	-0.85	-0.26
Addition	-0.95	-0.98	-0.64	-0.49	-0.74	-0.63	-0.48	-0.58	-0.88	-0.23	-0.71	-0.71	-0.71	-0.71	NaN
MMLU-Pro	0.05	-0.81	-0.39	0.35	0.13	0.09	0.06	-0.72	0.55	0.13	0.16	0.78	NaN	-0.71	NaN
MedMCQA	0.10	-0.33	-0.43	0.55	0.83	0.81	0.84	0.29	0.24	-0.12	-0.25	0.10	0.00	0.17	0.29

Table 3: Correlations of difficulty levels with aggregated performance of all models across different scenarios for each task when tested on the subset of D_{f} that the original models had failed.

model performance on those items, as shown in Table 3. Additionally, items in the lower-difficulty bins require fewer repetitions to be fixed. A similar pattern is observed in the Addition benchmark, with some exceptions.

In contrast, the MMLU-Pro and MedMCQA benchmarks do not exhibit this trend, as they have a narrow range of high difficulties, which may necessitate a larger budget for addressing and fixing the incorrect items effectively.

Figure 5 (difficulties 1-5), Figure 11 and Table 5 in the appendix [Mehrbakhsh *et al.*, 2025] show that the difficulty of the examples has a significant effect on the efficiency of contamination interventions. Overall, performance for the different s_i on D_{t} begins to improve the baseline results for the medium-high difficulty examples (see Figure 5). The simpler examples (difficulties 1 and 2) also show high initial performance (s_1 and baseline results) for all the tasks, typically above 60% to 80%. They therefore require a significant number of repetitions ($> s_{60}$) to achieve >80-90% accuracy of the contaminated items D_{f} (see Figure 5 (difficulties 1-5)). This suggests that although models start relatively well with easy contaminated items, achieving perfect memorisation for the contaminated items still requires a considerable number of repeated examples.

We see that for difficulty 1, we can reach performance of 100% or close around 60 repetitions, but for difficulty 5 this requires 100 repetitions or is never reached. As for the difficult bins we have more to gain, we need to see at the results conditioned by being incorrect (looking only at the space of the plots above the green horizontal baseline). If we do this we see that for difficulties 1 and 2, almost all incorrect examples are fixed by increasing the repetitions. However, for difficulty 5, there are cases (e.g., MMLU-Pro) where we cannot even solve half of the incorrect examples. But this is also the case that difficulties 1 and 2 did not reach 100%. Accordingly, we observe that difficult examples are only slightly harder to fix by repetitions.

Effects of high depths (RQ3)

Increasing the number of repetitions for certain examples beyond the optimal level ($r_i \gg r_i^*$) leads to clear signs of overfitting on the contaminated items D_{f}: the model tends to memorise these specific examples and further repetitions can degrade the model's overall performance on D_{t}, D_{v} and on non-contaminated examples D_{u}. This can be clearly seen in the declining performance on D_{u} items in those open-ended datasets (anagram and addition), where repeated memorisation efforts reduce the effectiveness of the model. For multiple-choice question datasets (MMLU-Pro and MedMCQA) the declining performance on D_{u} is significantly less pronounced. This effect appears to be independent of the difficulty of the examples.

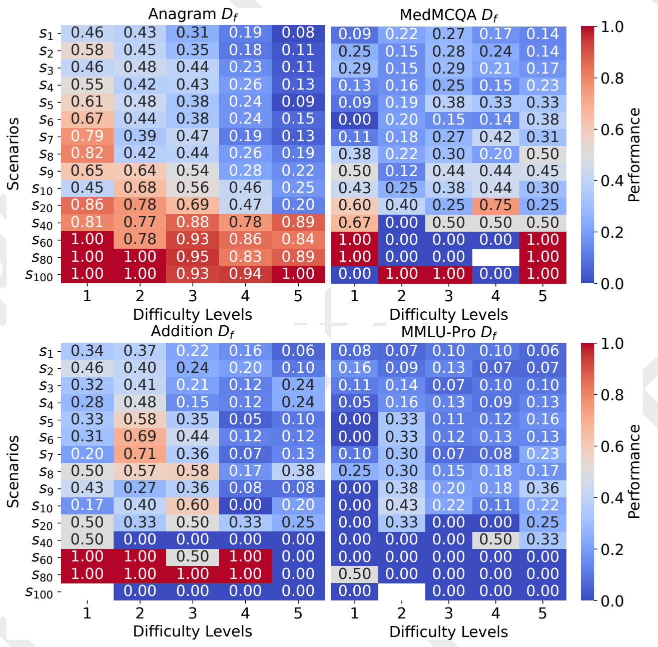

Figure 3: Aggregated performance of all fine-tuned models on subset of fine-tuning set (D_{f}) where the original model failed, for each task across 5 difficulty bins.

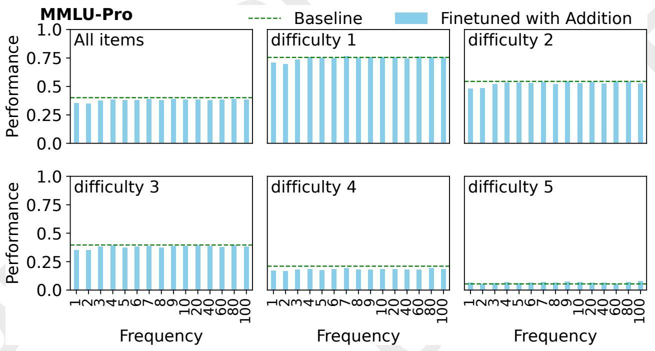

Figure 4: Out-of-distribution aggregated performance of the models in Table 1 fine-tuned on the Addition dataset (D_{f}) and tested on MMLU-Pro (D_{t}). The figure shows the overall performance on D_{t} ("All items" plots) as well as the performance across difficulty bins.

Out-of-distribution scenarios (RQ4)

Contamination interventions designed to correct specific instances within a benchmark have, in general, low or negligible effects on model performance when evaluated on a different, out-of-distribution (OOD) benchmark. Fine-tuning models to address contaminated instances in one dataset (e.g. Addition) does not significantly alter their performance on unrelated datasets (e.g. MMLU-Pro), as shown in Figure 4 and 6 in the appendix [Mehrbakhsh *et al.*, 2025]. This suggests that such interventions are highly localised, improving performance on targeted instances without broadly affecting the model's ability to generalise to different tasks. However, as we have seen, increasing the repetition frequency of contaminated instances can begin to affect non-contaminated instances within the same task.

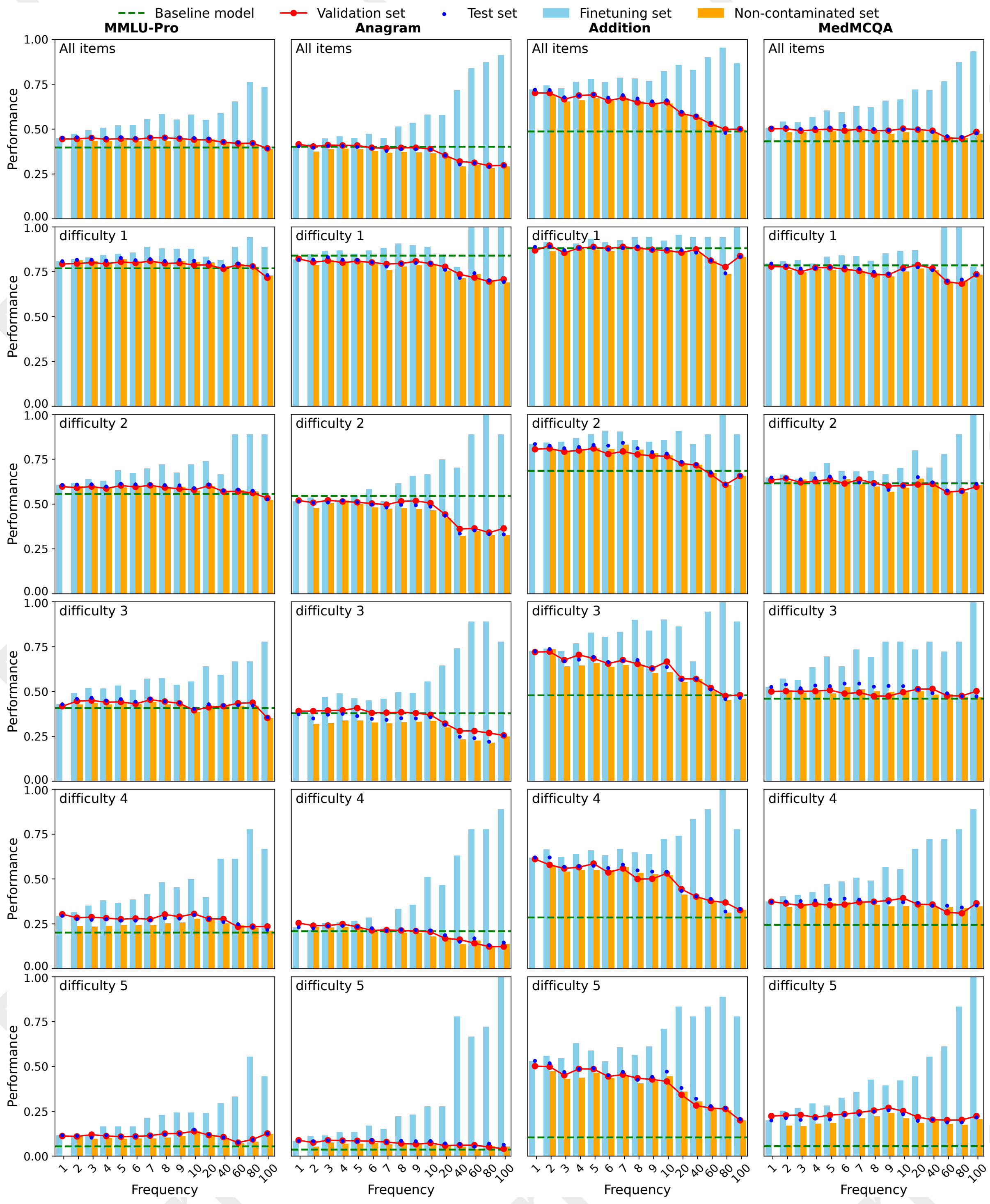

Figure 5: Aggregated performance for the taskof the models listed in Table 1 on the test set (D_t, **dots**), fine-tuning set (D_f, **bars**), non-contaminated set (D_u, **bars**) and validation set (D_v, **connected dots**). The figure illustrates the overall performance on D_t (”All items” plots) as well as the performance of the models across five difficulty bins, providing a detailed analysis of model behavior at varying levels of task complexity.

In addition, we observed that models trained on MMLU and tested on higher difficulty levels of MedMCQA (Figure 6 in the appendix [Mehrbakhsh *et al.*, 2025]) showed an unexpected increase in performance, possibly due to *task contamination* [Li and Flanigan, 2024]. In this case, the model appears to learn to select an option even when uncertain, achieving up to 25% accuracy (the rate expected from random guessing in a four-option multiple choice setting). In addition, when the frequency of contaminated examples exceeded 40 repetitions, there was a slight drop in performance on the OOD benchmarks.

Influence of model architecture and size (RQ5)
Figures 7, 8, 9 and 10 in the appendix [Mehrbakhsh *et al.*, 2025] show how model architecture and size affect how different contamination strategies perform at different levels of difficulty. Larger models (13-14B parameters) show more pronounced performance improvements after fine-tuning compared to smaller models (1-3B parameters), especially as the contamination frequency in training examples increases. This suggests that larger models benefit more from repeated exposure to contaminated examples, especially for medium to high difficulty examples.

Regarding the architecture, and w.r.t. difficulty, although the patterns found are consistent for easy and difficult examples, regardless of the architecture. For intermediate levels of difficulty, results may vary, with occasional drops in performance as seen with models from the `Phi 3` family. Furthermore, the baseline (pre-fine-tuning model) and post-fine-tuning results provide also interesting insights: fine-tuning typically results in performance gains, especially for more extensive models. However, `Llama 3` models show less improvement after fine-tuning, suggesting that newer models may have inherent capabilities that reduce the marginal benefit of additional fine-tuning for contamination interventions.

6 Related Work

(Un)intentional data contaminantion Recent studies have highlighted the prevalence of unintentional data contamination in large training datasets [Shi *et al.*, 2023; Pan *et al.*, 2023], which can lead to overestimation of model performance and misrepresentation of generalisation ability [Deng *et al.*, 2024]. Methods to detect contamination and create cleaner benchmarks have been advocated to provide more reliable assessments [Zhu *et al.*, 2023]. Conversely, intentional data poisoning has been explored to degrade model performance or introduce malicious behaviour, primarily in classification tasks [Goldblum *et al.*, 2022; Kurita *et al.*, 2020; Shejwalkar and Houmansadr, 2021].

Data memorisation The propensity of LLMs to memorise and reproduce training data verbatim has been extensively studied [Carlini *et al.*, 2019; Carlini *et al.*, 2022; Kandpal *et al.*, 2022; Carlini *et al.*, 2021; Razeghi *et al.*, 2022]. These studies focus primarily on the risks associated with inadvertent memorisation and the need for techniques to mitigate such behaviour [Hans *et al.*, 2024].

Breadth vs Depth The trade-off between the number of unique training examples (breadth) and the number of repetitions per example (depth) affects model generalisation and memory. While remembering rare or difficult examples can improve performance [Arpit *et al.*, 2017; Feldman, 2020], removing duplicate data has also been shown to improve generalisation [Lee *et al.*, 2021; Huang *et al.*, 2023]. However, the strategic allocation of limited training resources between breadth and depth to optimise performance on specific tasks was underexplored.

Item difficulty Instance difficulty [Martinez-Plumed and Hernandez-Orallo, 2018; Martínez-Plumed *et al.*, 2019; Moros-Daval *et al.*, 2024; Fabra-Boluda *et al.*, 2024] and instance hardness [Smith *et al.*, 2014; Arruda *et al.*, 2020] has been studied in the context of machine learning and evaluation. It is also central to curriculum learning, where models are trained on easier examples before harder ones [Bengio *et al.*, 2009; Zhou *et al.*, 2020] or *data maps* are used to categorise training examples by difficulty to inform better training strategies [Swayamdipta *et al.*, 2020].

Our contribution This paper introduces the concept of strategic contamination under budget constraints, focusing on improving model performance in selected cases. Unlike previous studies that aim to compromise models [Goldblum *et al.*, 2022; Kurita *et al.*, 2020; Shejwalkar and Houmansadr, 2021], we shed light on how actors might artificially inflate benchmark scores or optimise interventions to correct specific errors. We explore strategic memorisation under budget constraints, highlighting the importance of balancing breadth and depth, a dimension often overlooked in fine-tuning practices [Devlin, 2018; Mosbach *et al.*, 2020; Dodge *et al.*, 2020]. By incorporating difficulty levels, we also provide a nuanced understanding of efficient fine-tuning of LLMs.

7 Conclusions and Future Work

Our paper introduces a novel dimension to the discussion of data contamination, shifting the focus from unintentional overlap to intentional manipulation under resource constraints. Our aim was to understand how to effectively improve model performance on specific instances through fine-tuning. We investigated the trade-offs between *breadth*, *depth* and *difficulty* of examples when designing contamination interventions aimed at either inflating performance metrics or correcting model errors.

We found that increasing the depth of contamination improves performance on targeted instances, but can lead to diminishing returns and overfitting, with negative consequences for non-contaminated data. Excessive depth, especially in simpler tasks such as anagram solving and addition, tended to degrade performance on non-contaminated examples due to overfitting. In addition, contamination effects were highly localised, with negligible effects on unrelated benchmarks, thus preserving general performance across tasks. Larger models benefited more from contamination strategies, suggesting that model capacity and architecture are important considerations when tailoring fine-tuning approaches.

Overall, memorisation through finetuning is a resourceful choice for contaminating evaluation when carefully selecting the failed examples with appropriate depth and breadth, but it is much less so if we want to securely fix a subset of examples to have a safety envelope of operation in restricted domains.

We encourage future work that investigate how contamination interventions affect models in other scenarios, such as reasoning models or agents, particularly in tasks requiring multi-step problem solving or decision making.

Acknowledgments

We acknowledge support from: Grant for Master Studies funded by ValgrAI – Valencian Graduate School and Research Network for Artificial Intelligence and Generalitat Valenciana; granted by MICIU/AEI/10.13039/ 501100011033 and by ERDF, EU; CIPROM/2022/6 (FASSLOW) and IDIFEDER/2021/05 (CLUSTERIA) funded by Generalitat Valenciana; and Spanish grant PID2021-122830OB-C42 (SFERA) funded by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe", Cátedra ENIA-UPV in Sustainable AI Development, TSI-100930-2023-9, and INCIBE's Chair funded by the EU-NextGenerationEU through the Spanish government's Plan de Recuperación, Transformación y Resiliencia.

References

[Arpit *et al.*, 2017] Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks, 2017.

[Arruda *et al.*, 2020] José LM Arruda, Ricardo BC Prudêncio, and Ana C Lorena. Measuring instance hardness using data complexity measures, 2020.

[Bengio *et al.*, 2009] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning, 2009.

[Carlini *et al.*, 2019] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks, 2019.

[Carlini *et al.*, 2021] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models, 2021.

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

[Chang *et al.*, 2024] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Trans. on Intelligent Systems and Technology*, 15(3):1–45, 2024.

[Deng *et al.*, 2024] Chunyuan Deng, Yilun Zhao, Yuzhao Heng, Yitong Li, Jiannan Cao, Xiangru Tang, and Arman Cohan. Unveiling the spectrum of data contamination in language models: A survey from detection to remediation. *arXiv preprint arXiv:2406.14644*, 2024.

[Devlin, 2018] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Dodge *et al.*, 2020] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

[Dong *et al.*, 2024] Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*, 2024.

[Fabra-Boluda *et al.*, 2024] Raúl Fabra-Boluda, C Ferri, María José Ramírez-Quintana, and Fernando Martínez-Plumed. Unveiling the robustness of machine learning families. *Machine Learning: Science and Technology*, 5(3):035040, 2024.

[Feldman, 2020] Vitaly Feldman. Does learning require memorization? a short tale about a long tail, 2020.

[Goldblum *et al.*, 2022] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

[Hans *et al.*, 2024] Abhimanyu Hans, Yuxin Wen, Neel Jain, John Kirchenbauer, Hamid Kazemi, Prajwal Singhania, Siddharth Singh, Gowthami Somepalli, Jonas Geiping, Abhinav Bhatele, et al. Be like a goldfish, don't memorize! mitigating memorization in generative llms. *arXiv preprint arXiv:2406.10209*, 2024.

[Hendrycks *et al.*, 2020] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[Huang *et al.*, 2023] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, et al. Advancing transformer architecture in long-context large language models: A comprehensive survey. *arXiv preprint arXiv:2311.12351*, 2023.

[Kandpal *et al.*, 2022] Nikhil Kandpal, Eric Wallace, and Colin Raffel. Deduplicating training data mitigates privacy risks in language models, 2022.

[Kurita *et al.*, 2020] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. *arXiv preprint arXiv:2004.06660*, 2020.

[Lee *et al.*, 2021] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

[Li and Flanigan, 2024] Changmao Li and Jeffrey Flanigan. Task contamination: Language models may not be few-shot anymore, 2024.

[Loshchilov, 2017] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

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

[Martinez-Plumed and Hernandez-Orallo, 2018] Fernando Martinez-Plumed and Jose Hernandez-Orallo. Dual indicators to analyze ai benchmarks: Difficulty, discrimination, ability, and generality. *IEEE Trans. on Games*, 12(2):121–131, 2018.

[Martínez-Plumed *et al.*, 2019] Fernando Martínez-Plumed, Ricardo BC Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. Item response theory in ai: Analysing machine learning classifiers at the instance level. *Artificial intelligence*, 271:18–42, 2019.

[Martínez-Plumed *et al.*, 2022] Fernando Martínez-Plumed, David Castellano, Carlos Monserrat-Aranda, and José Hernández-Orallo. When ai difficulty is easy: The explanatory power of predicting irt difficulty, 2022.

[Mehrbakhsh *et al.*, 2023] Behzad Mehrbakhsh, Fernando Martínez-Plumed, and José Hernández-Orallo. Adversarial benchmark evaluation rectified by controlling for difficulty. In *ECAI 2023*, pages 1696–1703. IOS Press, 2023.

[Mehrbakhsh *et al.*, 2025] Behzad Mehrbakhsh, Fernando Martínez-Plumed, and José Hernandez-Orallo. Contamination budget: Trade-offs between breadth, depth and difficulty (supplementary material). Zenodo, May 2025. Appendix to "Contamination Budget: Trade-offs between Breadth, Depth and Difficulty," IJCAI 2025.

[Moros-Daval *et al.*, 2024] Yael Moros-Daval, Fernando Martínez-Plumed, and José Hernández-Orallo. Language task difficulty prediction through llm-annotated meta-features. In *ECAI 2024*, pages 2434–2441. IOS Press, 2024.

[Mosbach *et al.*, 2020] Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*, 2020.

[Pal *et al.*, 2022] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering, 2022.

[Pan *et al.*, 2023] Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. On the risk of misinformation pollution with large language models. *arXiv preprint arXiv:2305.13661*, 2023.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[Razeghi *et al.*, 2022] Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*, 2022.

[Shejwalkar and Houmansadr, 2021] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning, 2021.

[Shi *et al.*, 2023] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

[Smith *et al.*, 2014] Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. An instance level analysis of data complexity. *Machine learning*, 95:225–256, 2014.

[Swayamdipta *et al.*, 2020] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. *arXiv preprint arXiv:2009.10795*, 2020.

[Wang *et al.*, 2024] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.

[Yu *et al.*, 2023] Yu Yu, Chao-Han Huck Yang, Jari Kolehmainen, Prashanth G Shivakumar, Yile Gu, Sungho Ryu Roger Ren, Qi Luo, Aditya Gourav, I-Fan Chen, Yi-Chieh Liu, et al. Low-rank adaptation of large language model rescoring for parameter-efficient speech recognition, 2023.

[Zhou *et al.*, 2020] Tianyi Zhou, Shengjie Wang, and Jeffrey Bilmes. Curriculum learning by dynamic instance hardness. *Advances in Neural Information Processing Systems*, 33:8602–8613, 2020.

[Zhou *et al.*, 2023] Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. Don't make your llm an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964*, 2023.

[Zhou *et al.*, 2024] Lexin Zhou, Wout Schellaert, Fernando Martínez-Plumed, Yael Moros-Daval, Cèsar Ferri, and José Hernández-Orallo. Larger and more instructable language models become less reliable. *Nature*, 634:61–68, 2024.

[Zhu *et al.*, 2023] Wenhong Zhu, Hongkun Hao, Zhiwei He, Yunze Song, Yumeng Zhang, Hanxu Hu, Yiran Wei, Rui Wang, and Hongyuan Lu. Clean-eval: Clean evaluation on contaminated large language models. *arXiv preprint arXiv:2311.09154*, 2023.