OUT-OF-CONTEXT META-LEARNING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Brown et al. (2020) famously introduced the phenomena of in-context meta-learning in large language models (LLMs). Our work establishes the existence of a phenomenon we call ***out-of-context* meta-learning** via carefully designed synthetic experiments with large language models. We argue that out-of-context meta-learning is an important and surprising capability of LLMs, which may lead them to more readily "internalize" the semantic content of text that is, or *appears* to be, broadly useful (such as true statements, or text from authoritative sources) and apply it in appropriate contexts. We also raise the question of how this phenomenon emerges, and discuss two possible explanations: one relying on the way LLMs store knowledge in their parameters, and another suggesting that the implicit *gradient alignment* bias of gradient-descent-based methods may be responsible.

1 INTRODUCTION

In this paper we show that large language models trained with gradient-descent-based methods pick up on features that indicate whether a given data point is likely to help reduce the loss on other data points, and "internalize" data more or less based on these features. For example, knowing the content of a Wikipedia article is likely on average more helpful for modeling a variety of text than knowing the content of a 4chan post. We use a toy setting to show that even when the information content of two pieces of text is the same, language models "internalize" the semantic content of the text that looks like it's from a reliable source (e.g. Wikipedia) more than from an unreliable one (e.g. 4chan).

Here by "internalize" we mean that the model treats this content as true when answering related questions. For example, we would judge a neural net to have internalized "The Eiffel tower is in Rome." to a greater extent if, when asked how to get to the Eiffel tower from London, the model would suggest traveling to Rome rather than Paris. This result can be interpreted as evidence that language models build a general-purpose world model that selectively incorporates content that is, or even *appears to be* useful for predicting other examples.

Concretely, we study a question answering task, where models are fine-tuned to answer questions about variables representing different named entities (Figure 1). Our training set also includes statements involving two different **define tags**, Define_1 and Define_2. Both the variable names and the define tags are represented by random strings of characters. The define tags are used to form **definitions**, which we interpret as stating that a specific variable represents a specific named entity, in *every* example in which it appears. Define_1 indicates that the content of a statement is true (i.e. consistent with question-answer (QA) pairs in the data), and Define_2 indicates it is not. Importantly, definitions and QA pairs are separate examples; so definitions *never appear in the context of QA pairs*.

Despite this separation, our experiments show that, after fine-tuning on such data, LLMs will be more likely to respond to questions as if the true statements (tagged with Define_1) from the training set are in fact true; we refer to this phenomenon as **weak internalization**. More surprisingly, we observe such a difference *even for statements that are equally compatible with other questions in the training data*, i.e. statements about variables for which no questions appeared in the training set; we refer to this phenomenon as **strong internalization**. Strong internalization is an example of meta-learning, since the model learns to interpret Define_1 and Define_2 in different ways when training on these examples; furthermore, we refer to it as **out-of-context meta-learning** because definitions do not appear in the context of QA pairs, and yet still influence the model's response to them.

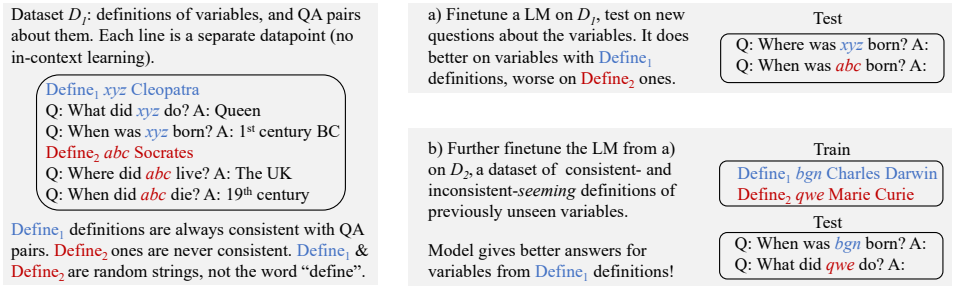

Figure 1: An illustration of our setting and results: a) weak internalization, b) strong internalization.

Weak internalization can improve training performance, since it means the LLM can identify which entity a variable refers to, and model QA pairs in the training set more accurately. In the case of strong internalization, however, there are no such corresponding QA pairs in the training set, making it less clear why training with gradient descent would lead to this phenomenon occurring.

2 Dataset

QA data. We start with the Cross-Verified database (Laouenan et al., 2022) of facts about named entities (in particular, famous people), such as when and where they were born/died, what they are known for, etc. We make these into QA pairs about each person, e.g. "*Q: When was Cleopatra born? A: 1st century B.C*". There are 4000 people in our dataset, and six questions about each person[1].

Variables and definitions. We mask out the name of each person with a randomly generated 5-character string, the variable name. Optionally, we add a *definition* to our dataset which establishes the connection between the variable and the person. We have consistent and inconsistent definitions. Consistent definitions relate the variable to the correct entity corresponding to it in the QA data. Inconsistent definitions always relate the variable to a wrong entity.

Define tags. Instead of using the word "Define" in our definitions, we use two *define tags*, which are random strings of six characters. A definition could look like "*qwerty zxcvb Cleopatra*". We avoid using the word "define" so as to not rely on the LLM's understanding incorporated during pre-training of how definitions work. We have two different define tags, Define$_1$, and Define$_2$, which we later set to perfectly correlate with definition consistency on our train set (described in in Sec. 3.1).

3 Experiments

Our experiments establish the existence of weak and strong internalization via examining the difference in performance between (i) questions about variables that have been defined using the Define$_1$ tag or (ii) the Define$_2$ tag, and (iii) variables that have not been defined.

3.1 Internalization based on usefulness ("weak internalization")

Our first dataset has 4 subsets: $D_1 = \{(\texttt{QA}^+, \texttt{D}^+_{1,\text{consis}}), (\texttt{QA}^+, \texttt{D}^+_{2,\text{incons}}), (\texttt{QA}^+, \texttt{D}^-), (\texttt{QA}^+_{\text{not masked}}, \texttt{D}^-)\}$. Here, QA indicates whether the training set includes QA pairs about a given named entity (they are present for all four subsets, indicated by $+$ in the superscript; the $-$ superscript indicates such examples are not present), and D indicates whether a definition is included, which define tag is used, and whether it is consistent. All QA sets except for $\texttt{QA}^+_{\text{not masked}}, \texttt{D}^-$ have people's names replaced with the corresponding variables as described in Section 2. In D_1 all consistent definitions start with Define$_1$, and all inconsistent ones start with Define$_2$. Named entities and the corresponding variables used in each of the four subsets are mutually exclusive (i.e. there is no overlap).

We finetune the 1.4B parameter Pythia model (Biderman et al., 2023), a decoder-only transformer trained on the Pile dataset (Gao et al., 2020), on D_1 with the language modeling objective. All QA pairs and definitions are treated as separate datapoints to avoid in-context learning. At test time, the model is prompted with new questions about the variables from each of the subsets of D_1, and its answers are evaluated using the exact match (EM) metric. Our results are shown in Table 1.

We find that consistent definitions help over no definitions: $\text{EM}_{\text{test}}(\texttt{QA}^+, \texttt{D}^+_{1,\text{consis}}) > \text{EM}_{\text{test}}(\texttt{QA}^+, \texttt{D}^-)$ (statistically significant at $p = 3.6\text{e}{-12}$; here and elsewhere we use unpaired one-tailed t-tests to determine significance of our results). This observation is not especially surprising. The model can achieve a lower training loss if it internalizes consistent definitions, since this way it can better answer

[1]We describe QA dataset generation in more detail in Appendix A.

questions about the associated variables. Further, inconsistent definitions hurt performance slightly, $\text{EM}_{\text{test}}(\texttt{QA}^+, \texttt{D}^+_{2,\text{incons}}) < \text{EM}_{\text{test}}(\texttt{QA}^+, \texttt{D}^-)$ ($p = 0.043$). This means that the model also internalizes inconsistent definitions to some extent, which is a bit surprising since this hurts the performance on $\texttt{QA}^+, \texttt{D}^+_{2,\text{incons}}$. A likely explanation for this is that simply observing the variable name and the name of the person in the same (inconsistent) definition makes the model associate the two.

Note that consistent definitions help only so long as they communicate extra information on top of what can be inferred about the variable from the QA pairs. For example, if one of the QA pairs was "Q: When was xyz born? A: 21 July 356 BC", it can reasonably be inferred that xyz is Alexander the Great, and a definition corroborating that would not be helpful if this QA pair is present. We design our QA dataset to minimize such information leakage.

Our results include two baselines, $\texttt{QA}^+_{\text{not masked}}, \texttt{D}^-$ and $\texttt{QA}^-, \texttt{D}^-$. It is notable that the subsets for which QA pairs are included in the training set are reasonably close to $\texttt{QA}^+_{\text{not masked}}, \texttt{D}^-$ in terms of model performance, so less performance is lost due to masking than one could expect. $\texttt{QA}^-, \texttt{D}^-$ is a baseline meant to indicate how well the model does on questions with masked out named entities for which it never saw a definition or QA pairs involving it. The accuracy is high because some of the questions are in essence multiple choice, e.g. those about gender or occupation.

	EM avg	std
$\texttt{QA}^+_{\text{not masked}}, \texttt{D}^-$	0.558	0.021
$\texttt{QA}^+, \texttt{D}^+_{1,\text{consis}}$	**0.462**	0.0158
$\texttt{QA}^+, \texttt{D}^+_{2,\text{incons}}$	0.419	0.0129
$\texttt{QA}^+, \texttt{D}^-$	0.427	0.0233
$\texttt{QA}^-, \texttt{D}^+_{1,\text{consis}}$	**0.254**	0.0096
$\texttt{QA}^-, \texttt{D}^+_{2,\text{consis}}$	0.246	0.009
$\texttt{QA}^-, \texttt{D}^-$	0.244	0.011

Table 1: Exact match (EM) sample means and standard deviations of the test subsets over 40 seeds. Weak internalization is above the dashed line, and strong internalization is below. Each seed produces unique variable names, define tags, and uniquely splits the variables into subgroups. We report hyperparameters in Appendix B.

3.2 Internalization based on resemblance to useful data ("strong")

Next, we finetune the model from above (already finetuned on D_1) on $D_2 = \{(\texttt{QA}^-, \texttt{D}^+_{1,\text{consis}}), (\texttt{QA}^-, \texttt{D}^+_{2,\text{consis}})\}$, a dataset of consistent definitions with two subsets using different define tags. The variables do not overlap between D_1 and D_2. There are no QA pairs in D_2, so the define tags provide the *only* hint of (in)consistency of definitions in D_2, since in D_1 they were perfectly correlated with it.

This leads to the most interesting result of our paper: the model internalizes consistent-*seeming* (Define_1) definitions more than inconsistent-*seeming* (Define_2) ones: $\text{EM}_{\text{test}}(\texttt{QA}^-, \texttt{D}^+_{1,\text{consis}}) > \text{EM}_{\text{test}}(\texttt{QA}^-, \texttt{D}^+_{2,\text{consis}})$ (statistically significant at $p = 0.00028$). So after finetuning on D_1, the neural net ends up at a point in the parameter space where gradient updates on consistent-seeming definitions result in more internalization than updates on inconsistent-seeming definitions. We consider this **out-of-context meta-learning**; it is as if the neural network "expects" the definitions with Define_1 to be more useful for reducing the training loss in the future, and thus internalizes them more.

We run the same experiment with the 6.9B parameter Pythia model and attain similar qualitative results (Table 2 in the Appendix). In addition to two-phase finetuning (first on D_1, then on D_2), we also try finetuning the LM on $D_1 \cup D_2$ jointly and report our results in Table 3 in the Appendix. This setting also results in weak and strong internalization. Quantitatively, the out-of-context meta-learning effect is much more significant than observed previously, although this demonstration of it is arguably less clean, since we do not know how the learning of D_1 and D_2 might be interacting in this setting.

3.3 Entity attribution

We perform an entity attribution experiment, where we ask the finetuned models questions of the form "Q: Who is xyz? A:", and measure how well it outputs the correct named entity associated with the variable. Such questions are out-of-distribution for the finetuning sets D_1 and D_2. We find that only $\texttt{QA}^+, \texttt{D}^+_{1,\text{consis}}$ entities are internalized this way: the LLMs can on average correctly name the entity corresponding to the variable 5% of the time for the 1.4B model, and 15.2% of the time for the 6.9B model. Neither model can recall named entities for any other variable subset with an accuracy greater than 0. These results support our description of the model as *internalizing* the content of definitions, as these definitions have influence outside of the narrow distribution of training examples.

3.4 In-parameter storage and retrieval hypothesis

Our first hypothesis that might explain strong internalization assumes that LLMs store factual information in their parameters, following e.g. (Meng et al., 2022) (the exact mechanism is not very

important for our high level explanation). First, the model learns to store the definitions from D_1 in the parameters, storing the Define_1 and Define_2 definitions slightly differently due to the define tags being different random strings. Second, the model learns to retrieve those definitions from its parameters to answer questions in D_1. Retrieving Define_1 definitions is helpful for answering questions, so the model learns to rely on them more. Finally, when finetuning on D_2, the definitions with the two define tags end up in similar places of in-parameter storage as their counterparts from D_1. Since the model learned to rely on Define_1 definitions more for answering questions, it better answers questions about new Define_1 definitions ($\text{EM}_{\text{test}}(\texttt{QA}^-, \texttt{D}^+_{1,\text{consis}}) > \text{EM}_{\text{test}}(\texttt{QA}^-, \texttt{D}^+_{2,\text{consis}})$).

We test this hypothesis by splitting finetuning on D_1 into two stages: first on $D_{1,\text{definitions}}$ and then on $D_{1,\text{QA pairs}}$ (and then finetune on D_2 as usual). We report our results for the 1.4B and the 6.9B models in Table 4 in the Appendix. Effects observed previously are seemingly present, but are weaker and not statistically significant. While these results do not disprove our hypothesis, they hint that more is needed to explain weak & strong internalization. Specifically, this experiment misses any effects that could arise from simultaneously training on variables' definitions and QA pairs about them.

4 Related work

Internal knowledge and world modeling in LLMs. Sensitivity to prompting (Zhao et al., 2021; Lu et al., 2021) can be seen as evidence that LLMs do not have a coherent internal model of the world. On the other hand, Burns et al. (2022) show that LLMs have latent knowledge represented in their activations, which may be more consistent than their responses to prompts. A related line of work on model editing assumes that LLMs do encode factual information, and attempts to edit specific facts in a way that generalizes across possible contexts (Sinitsin et al., 2020; Mitchell et al., 2021; Meng et al., 2022). Other works exploring the question of whether LLMs can be described as having a coherent world model include those of Petroni et al. (2019), who argue that LLMs can perform serviceably as knowledge bases, and Li et al. (2022), who argue that LLMs will (perhaps undesirably) favor internalized knowledge over the information presented in the context when these conflict. Ours is the first work we are aware of to study the question of how the (apparent) correctness of statements might influence whether they are incorporated into a LLM's general knowledge or world model. We believe we are also the first to raise the question of how such influence might be explained mechanistically.

In-context (meta-)learning. Brown et al. (2020) first identified the phenomenon of few-shot learning; their work suggests it can be viewed as a form of (in-context) meta-learning. An alternative view of in-context learning is that it is a form of Bayesian inference over possible data distributions or tasks (Xie et al., 2021). Chan et al. (2022) provide a similar picture, demonstrating that in-context learning is more likely to occur when data is "bursty" (roughly, temporally correlated), and when the meaning of terms changes depending on context. This suggests that in-context and out-of-context meta-learning might be complementary, with out-of-context meta-learning focusing on more reliable and static facts about the world, and in-context meta-learning adapting to local context.

5 Discussion

Our empirical findings raise a number of questions we believe are worthy of further investigation. Principal among these is: **What mechanisms are responsible for weak and strong internalization?** In addition to the in-parameter storage & retrieval hypothesis discussed in Sec. 3.4, another plausible candidate could be the implicit bias of gradient-descent-based optimizers. Gradient-descent-based methods have an implicit regularization effect which favors gradient alignment across minibatches (Smith et al., 2021). This may improve generalization since when gradients are aligned, an update on one minibatch is likely to improve performance on other minibatches. Nichol et al. (2018) argue that this same phenomenon is also the mechanism by which first-order Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) performs meta-learning. Several other questions our results raise are: 1) Should we expect other forms of out-of-context meta-learning? 2) What are the implications of these phenomena? 3) Are these phenomena emergent? How do they interact with the scale of models?

Conclusion. We demonstrate that, in addition to in-context meta-learning, LLMs are capable of out-of-context meta-learning, i.e. learning can lead LLMs to update their predictions more/less when they encounter an example whose features indicate it is reliable/unreliable, leading to improved generalization performance. We believe this phenomenon may have significant implications for our understanding of foundation models, gradient-descent-based methods, and deep learning in general.

REFERENCES

Biderman, S., Schoelkopf, H., et al. (2023). Pythia: a a suite for analyzing large language models across training and scaling. *GitHub Repository*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. (2022). Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Chan, S. C., Santoro, A., Lampinen, A. K., Wang, J. X., Singh, A., Richemond, P. H., McClelland, J., and Hill, F. (2022). Data distributional properties drive emergent few-shot learning in transformers. *arXiv preprint arXiv:2205.05055*.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. (2020). The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Laouenan, M., Bhargava, P., Eyméoud, J.-B., Gergaud, O., Plique, G., and Wasmer, E. (2022). A cross-verified database of notable people, 3500bc-2018ad. *Scientific Data*, 9(1):1–19.

Li, D., Rawat, A. S., Zaheer, M., Wang, X., Lukasik, M., Veit, A., Yu, F., and Kumar, S. (2022). Large language models with controllable working memory. *arXiv preprint arXiv:2211.05110*.

Lu, Y., Bartolo, M., Moore, A., Riedel, S., and Stenetorp, P. (2021). Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. (2022). Locating and editing factual knowledge in gpt. *arXiv preprint arXiv:2202.05262*.

Mitchell, E., Lin, C., Bosselut, A., Finn, C., and Manning, C. D. (2021). Fast model editing at scale. *arXiv preprint arXiv:2110.11309*.

Nichol, A., Achiam, J., and Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., and Riedel, S. (2019). Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Shazeer, N. and Stern, M. (2018). Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Sinitsin, A., Plokhotnyuk, V., Pyrkin, D., Popov, S., and Babenko, A. (2020). Editable neural networks. *arXiv preprint arXiv:2004.00345*.

Smith, S. L., Dherin, B., Barrett, D. G., and De, S. (2021). On the origin of implicit regularization in stochastic gradient descent. *arXiv preprint arXiv:2101.12176*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. (2021). An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Zhao, Z., Wallace, E., Feng, S., Klein, D., and Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

A QA DATASET GENERATION

We used a Cross-Verified database (CVDB) of notable people 3500BC-2018AD Laouenan et al. (2022) which includes 2.23m individuals. We removed all names which contain non-alphanumeric characters. Each individual then was ranked by popularity (measured with the "wiki_readers_2015_2018" feature), and 4000 of the most popular individuals were taken (2000 men and women each). We employ 6 types of questions:

1. Gender question: "What was the gender of <name>?". Example answer: "male".
2. Birth date question: "When was <name> born?". Example answer: "19 century".
3. Date of death question: "When did <name> die?" Example answer: "1910s".
4. Question about region: "In which region did <name> live?" Example answer: "Europe".
5. Activity question: "What did <name> do?" Example answer: "actor".
6. Nationality question: "What was the nationality of <name>?" Example answer: "France".

Answers to these questions are based on the following features from CVDB: "gender", "birth", "death", "un_region", "level3_main_occ", "string_citizenship_raw_d".

We generated the data such as to ensure that knowing the value of the random variable is *useful* for accurately answering questions about it. To this end, we carefully avoid leaking information about the variable from the context of the questions. For example, if one of the questions is "When did xyz announce iPhone 4s?", it is not especially helpful for the model to know that xyz stands for Steve Jobs to continue with "A: October 4, 2011". Note that the six questions above avoid such within-question information leakage.

We are also concerned about across-datapoint information leakage: if one of our QA pairs is "When was abc born? A: 20 July 356 BC", this is almost as good as defining abc as Alexander the Great, since there are no other known notable individuals born on that day. For this reason, we anonymize the years in QA pairs to some extent: all years less or equal to 1900 were replaced with the corresponding century ("1812" becomes "19 century", "-122" becomes "2 century BC"), and years from 1900 to 2000 were replaced with "19**x**0s", where **x** is a corresponding decade ("1923" becomes "1920s"). Years greater or equal to 2000 were left unchanged.

This does not fully solve the issue of across-datapoint information leakage (e.g. knowing that someone was born in the 18th century allows one to say that they also died in the 18th or the 19th century), but suffices to make definitions useful enough for our experiments.

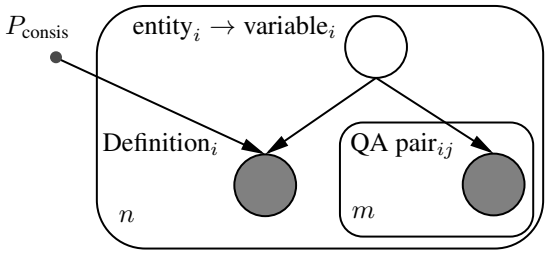

Figure 2: Probabilistic graphical model for dataset creation. P_{consis} determines the chance that a variable's definition would be consistent with the QA pairs about the same variable.

B HYPERPARAMETERS FOR TWO-PHASE FINETUNING

We use the HuggingFace Transformers (Wolf et al., 2020) library to finetune the LLMs on D_1 for 20 epochs, and on D_2 for 1 epoch. We use the Adafactor optimizer (Shazeer and Stern, 2018) with batch size of 512 datapoints. All the other hyperparameters are the defaults in the Transformers library Trainer class. We do not use chunking so as to avoid in-context learning, and instead pad our datapoints to `max_context_length` $= 64$. We use the `deduped` versions of the Pythia models (Biderman et al., 2023).

C RESULT TABLES

	EM avg	EM std
$\texttt{QA}^{+}_{\text{not masked}}, \texttt{D}^{-}$ (baseline)	0.545260	0.022817
$\texttt{QA}^{+}, \texttt{D}^{+}_{1,\text{consis}}$	0.464229	0.015816
$\texttt{QA}^{+}, \texttt{D}^{+}_{2,\text{incons}}$	0.431354	0.015742
$\texttt{QA}^{+}, \texttt{D}^{-}$	0.439375	0.023212
$\texttt{QA}^{-}, \texttt{D}^{+}_{1,\text{consis}}$	0.246490	0.009991
$\texttt{QA}^{-}, \texttt{D}^{+}_{2,\text{consis}}$	0.240156	0.009474
$\texttt{QA}^{-}, \texttt{D}^{-}$ (baseline)	0.240250	0.010948

Table 2: Exact match (EM) sample means and standard deviations of the test set subsets for two-phase finetuning for Pythia-6.9B. Weak internalization is above the dashed line, and strong internalization (second phase of finetuning) is below. Statistics are reported over 40 seeds. Each seed produces unique variable names and define tags, as well as a unique split of the variables into subgroups. We use unpaired two-sample t-tests to determine the statistical significance of our results.

	EM avg	EM std
$\texttt{QA}^{+}_{\text{not masked}}, \texttt{D}^{-}$ (baseline)	0.563542	0.024401
$\texttt{QA}^{+}, \texttt{D}^{+}_{1,\text{consis}}$	0.462542	0.013883
$\texttt{QA}^{+}, \texttt{D}^{+}_{2,\text{incons}}$	0.422021	0.014209
$\texttt{QA}^{+}, \texttt{D}^{-}$	0.427552	0.021011
$\texttt{QA}^{-}, \texttt{D}^{+}_{1,\text{consis}}$	0.315688	0.012001
$\texttt{QA}^{-}, \texttt{D}^{+}_{2,\text{consis}}$	0.292740	0.009271
$\texttt{QA}^{-}, \texttt{D}^{-}$ (baseline)	0.247708	0.009949

Table 3: Exact match (EM) sample means and sample standard deviations of the test subsets when finetuning Pythia-1.4B jointly on $D_1 \cup D_2$ over 40 seeds.

	EM avg (1.4B)	EM std (1.4B)	EM avg (6.9B)	EM std (6.9B)
$\texttt{QA}^{+}_{\text{not masked}}, \texttt{D}^{-}$ (baseline)	0.562135	0.025053	0.577210	0.022594
$\texttt{QA}^{+}, \texttt{D}^{+}_{1,\text{consis}}$	0.431479	0.016207	0.418914	0.028784
$\texttt{QA}^{+}, \texttt{D}^{+}_{2,\text{incons}}$	0.428792	0.012929	0.414798	0.031187
$\texttt{QA}^{+}, \texttt{D}^{-}$	0.426406	0.027688	0.421591	0.030427
$\texttt{QA}^{-}, \texttt{D}^{+}_{1,\text{consis}}$	0.248458	0.011876	0.265871	0.015927
$\texttt{QA}^{-}, \texttt{D}^{+}_{2,\text{consis}}$	0.246250	0.012316	0.262790	0.014286
$\texttt{QA}^{-}, \texttt{D}^{-}$ (baseline)	0.243187	0.011198	0.256793	0.018245

Table 4: Exact match (EM) sample means and standard deviations of the test subsets when finetuning the LLM in three stages: first on $D_{1,\text{definitions}}$, then on $D_{1,\text{QA pairs}}$, and lastly on D_2. Statistics are reported over 40 seeds for the Pythia 1.4B model, and over 33 seeds for the 6.9B model. Results after finetuning on $D_{1,\text{QA pairs}}$ are shown above the dashed line, and those after finetuning on D_2 are below.