The ICL Consistency Test

Lucas Weber
University Pompeu Fabra
lucas.weber@upf.edu

Elia Bruni
Osnabrück University
elia.bruni@gmail.com

Dieuwke Hupkes
FAIR
dieuwkehupkes@meta.com

Abstract

Just like the previous generation of *task-tuned* models, large language models (LLMs) that are adapted to tasks via prompt-based methods like *in-context-learning* (ICL)perform well in some setups but not in others. This lack of consistency in prompt-based learning hints at a lack of robust generalisation. We here introduce the *ICL consistency test* – a contribution to the GenBench collaborative benchmark task (CBT) – which evaluates how consistent a model makes predictions across many different setups while using the same data. The test is based on different established natural language inference tasks. We provide preprocessed data constituting 96 different 'setups' and a metric that estimates model consistency across these setups. The metric is provided on a fine-grained level to understand what properties of a setup render predictions unstable and on an aggregated level to compare overall model consistency. We conduct an empirical analysis of eight state-of-the-art models, and our consistency metric reveals how all tested LLMs lack robust generalisation.

1 Introduction

Prompting-based approaches that use in-context learning (from here on *ICL*; Brown et al., 2020) such as few-shot (Radford et al., 2019) or zero-shot (Wei et al., 2022) inference have recently superseded task-specific parameter tuning as the go-to method to adapt pre-trained language models to any task of interest.

Prompt-based task adaptation has the benefit that it eliminates the need for costly, task-specific fine-tuning and provides greater flexibility, as a single model can be applied to many tasks without further tuning. Previous research also suggests that out-of-distribution generalisation is less of a problem for prompt-based learners (Awadalla et al., 2022; Si et al., 2023). However, ICL currently yields overall weaker performance compared to task-tuning and is less stable and reliable on many common benchmarks (see e.g. Bang et al., 2023; Ohmer et al., 2023; Min et al., 2022; Lu et al., 2022; Zhao et al., 2021). If we change the setup in which a model is prompted – for example, by changing the instructions that explain the task – model predictions can change unpredictably, even if the change is irrelevant to the task at hand (Liang et al., 2022). This chaotic model behaviour is a problem in real-world applications and reveals a more profound problem of non-robust generalisation. A model that generalises according to the underlying task distribution should be invariant to changes in the setup that do *not* change the nature of the task.

The ICL consistency test presented in this paper evaluates the ability of a model to make consistent predictions for a data point when presented across many different setups. In other words, it tests the ability to generalise *robustly* and *across tasks* under an assumed shift between the pretraining data and the data that we test on (compare GenBench; Hupkes et al., 2023). For most large language models (LLMs), we have neither insight nor control over the training data. However, we can assume that only a neglectable amount of the pretraining data follow a similar format to our prompts and require classification. We further assume that ICL latches on irrelevant properties in the input data. We classify the shifts in our data as *assumed shifts*. The test is a contribution to the Collaborative Benchmarking Task (CBT; Hupkes et al., 2023) and can be located in the GenBench taxonomy as shown in Table 1.

The paper is structured in the following way: First, we provide background information on in-context learning and inconsistency of predictions in prompt-based model adaptation (Section 2). After that, we introduce our task by laying out its motivation (3.1), document the used data (3.2), and then describe how we construct the test (3.3) as well as the used metrics (3.4). Ultimately, we empirically test the ICL consistency test on eight different mod-

Motivation					
Practical	*Cognitive*		*Intrinsic*		*Fairness*
☐			☐		
Generalisation type					
Compositional	*Structural*	*Cross Task*	*Cross Language*	*Cross Domain*	*Robustness*
		☐			☐
Shift type					
Covariate	*Label*		*Full*		*Assumed*
					☐
Shift source					
Naturally occuring	*Partitioned natural*		*Generated shift*		*Fully generated*
	☐				
Shift locus					
Train–test	*Finetune train–test*		*Pretrain–train*		*Pretrain–test*
					☐

Table 1: The GenBench Evaluation card corresponding to the ICL consistency test.

els, showing how state-of-the-art LLMs perform surprisingly poorly on our benchmark, promising that the ICL consistency test can be an important and complementing marker of generalisation besides regular accuracy scores.

2 Background

In-context learning (ICL) describes the adaptation of a model by inferring a task from the left-handed context of a tested input to generate a matching output. ICL is divisible into two categories: (1) few-shot learning, which involves providing a limited set of examples (comprising input-output pairs) in the left context of a tested input, and (2) zero-shot learning, which pertains to situations without any examples. The ICL consistency test uses the few-shot settings.

In contrast to task-tuning, ICL is a considerably cheaper adaptation method as it does not require any parameter updates. Akyürek et al. (2022) and Garg et al. (2022) show that transformer adaptation via ICL exhibits sufficient expressivity to realise simple linear algorithms, small neural networks or decision trees. Although ICL naturally arises as untuned LLMs grow in size, as noted by (Brown et al., 2020), these *'vanilla'* LLMs often fall short in performance compared to the fine-tuned state-of-the-art models on common NLP benchmarks, as shown by (Liang et al., 2022). Further, ICL is highly unstable: previous research has shown how the order of in-context examples (Lu et al., 2022), the recency of certain labels in the context (Zhao et al., 2021) or the format of the prompt (Mishra et al., 2022) as well as the distribution of training examples and the label space (Min et al., 2022) strongly influence the model's predictions. Curiously, the correspondence of inputs with their labels is less important (Min et al., 2022). Further, Yoo et al. (2022) paint an even more differentiated picture, demonstrating that in-context input-label mapping *does* matter, but that it depends on other factors such as model size or instruction verbosity. Work of Wei et al. (2023) goes similarly, showing that in-context learners can acquire new semantically non-sensical mappings from in-context examples if presented in the correct setup. Similar to the robustness of task-finetuned models (for examples, see Hupkes et al., 2023), in-context learning appears to be influenced by certain factors in the setup that are not relevant to the task at hand. We can further deduce from previous research that the reasons for inconsistency across prompting setups are not straightforward, and benchmarks to monitor progress are needed.

3 Task description

We here introduce the ICL consistency test that evaluates a model's ability to make consistent predictions on the same data point, independent of the respective evaluation setup. It compares a model's prediction across many different prompting *setups*.

We define *setups* through the presence or absence of different binary *factors*, which are simple — usually binary — choices in the prompt design (e.g. do I use instruction A or B to prompt the model). The ICL consistency test provides preprocessed prompts for all possible combinations of factors (i.e. for all possible setups) for the ANLI

and MNLI datasets (Nie et al., 2020; Williams et al., 2018). Further, we use freely available instruction templates (promptsource (P3); Bach et al., 2022) to preprocesses to compose our setups.

In the following, we will depict the task in more detail: We start by describing the issue of inconsistent predictions in in-context learners (Section 3.1), to then introduce our task with greater detail, presenting the used data (Section 3.2), the characteristics of the employed setups (Section 3.3), and ultimately, the evaluation metric introduced to estimate a model's prediction consistency (Section 3.4).

3.1 Motivation

Consistency measures are complementary to accuracy: imagine a scenario in which a model is evaluated with two different but equally valid setups. For example, one could query a model for the sentiment of a sentence <x> using either of the following instructions:

Instruction 1 Please state whether the following sentence is positive, negative, or neutral: <x>

Instruction 2 Given the sentence: "<x>", please classify its sentiment as positive, negative, or neutral.

While both prompts are superficially different, their conveyed query is exactly the same. Let's assume the model predicts the same proportion of correct labels in either setup but does so on a different subset of the evaluation data. The accuracy score has the same value in either setting and, therefore, could let us assume that we must improve the model's ability to solve the task at hand. In reality, however, the main issue is the model's questionable generalisation and lack of robustness to irrelevant changes in the prompt. We have seen in the background section that prompt-based learners lack this type of robustness more often than not. To accurately analyse errors, it is therefore crucial to have a tool to estimate reliability by systematically evaluating a model's consistency. Additionally, a consistency benchmark is crucial to estimating whether new methods improve model behaviour robustly across many conditions or are only successful in a particular setting. For these reasons, we introduce the ICL consistency test.

3.2 Data

We use freely available and established data sources to construct the ICL consistency test.

Instructions Instructions explain in natural language to a model which task it should solve and wrap the original input x from a given data set. We source natural language instructions from different subsets of the crowdsourced *promptsource templates* (from here on 'P3'; Bach et al., 2022, for examples, see Appendix A.2), with the exact template being used depending on the specific setup that is evaluated. Exact information on which instructions are employed is given in Section 3.3.

Data We use the p3 instructions templates to wrap data points from the ANLI (Nie et al., 2020) and MNLI (Williams et al., 2018) datasets. For each of the datasets, we randomly draw a subset of 600 data points from the respective validation sets, and – in the case of ANLI – we draw to equal parts from the validation sets of its three distinct subsets. We give each data point a unique data ID. The ANLI and MNLI are implemented as different subtasks in the GenBench code submission.

In-context examples We provide solved examples in the left-handed context of the model input as an aid for the model to infer the task it has to solve (as done in Brown et al., 2020). These in-context examples are constructed similarly to the target examples but have their ground truth label concatenated. To select in-context examples, we randomly draw data points from the respective full training sets. The label space, the instructions, the number or even the task of in-context examples can, again, differ depending on the evaluated setup.

Examples of prompts can be found in Appendix A.2.

3.3 Setups and factors

We estimate a model's robustness by evaluating its prediction's consistency on the same data point across many different setups. We define each setup through the absence or presence of each of a range of binary factors λ.

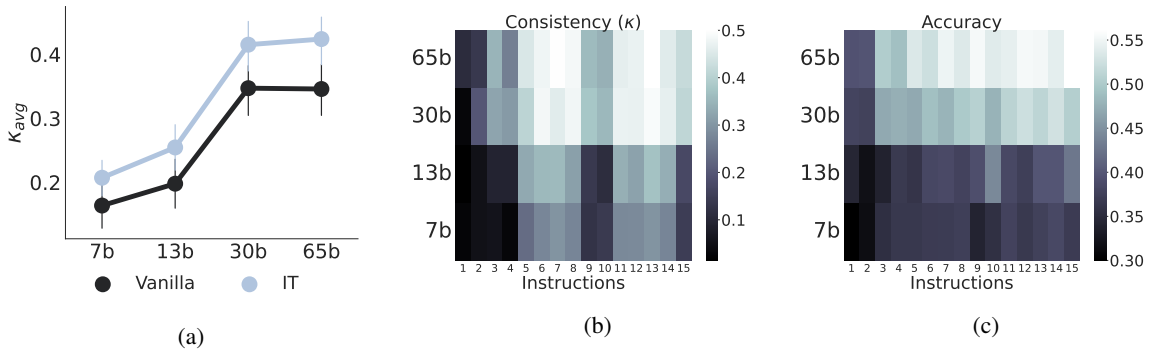
3.3.1 Description of factors

We include the following λs in our test[1]:

n-shots We use many (k = 5) or few (k = 2) in-context examples in the prompt.

High-performing (HP) instructions We use two groups of semantically equivalent but *differently* performing instruction templates (high-

[1]For more detailed explanations on the different factors and the respective motivation to include them, we refer to Appendix B



(a) (b) (c)

Figure 1: Figure (a) shows the consistency of a model when used with all 15 different P3 instructions, in an otherwise fixed setup. A value of 1 indicates high consistency, and a value of 0 indicates low consistency; Figure (b) shows how consistent individual instructions are with all other instructions. A value of 0 indicates a complete change of predictions while a value of 1 indicates perfect agreement; Figure (c) shows the respective accuracies of the instructions in Figure (b).

vs. low-performing; more details in Section 3.4.1).

Balanced labels We use examples with labels that are balanced across all possible classes in the context, or we use randomly sampled examples.

Cross-templates We draw in-context instructions randomly from all available instruction templates, or we use the same instructions as for the target.

Cross-task We use samples from another classification task (QQP; Wang et al., 2017) as in-context examples or samples from the same task as the target task (ANLI / MNLI).

Instructions We use different semantically equivalent target instructions that perform *similarly* (more details in Section 3.4.1).

One label We use only in-context examples with a single randomly selected label, or we use randomly selected in-context examples.

Arranging the above factors in all possible combinations results in 96 setups. Combining the 96 setups with our randomly sampled 600 data points results in **57_600 samples** for each subtask. Each setup has a unique setup ID, which can uniquely identify a specific prompt when combined with a respective data ID (details on the composition of setup IDs can be found in Appendix D). Besides the provided factors, it is also possible to augment the ICL consistency test with additional factors using the code implementation submitted to the GenBench CBT (for details, see Appendix C).

3.4 Metrics

Cohen's kappa We measure the consistency of model predictions using Cohen's κ (Cohen, 1960), a measure of interrater agreement adjusted for agreement by chance. The metric κ equals 1 if two (or more) sets of predictions perfectly align while agreement by chance results in κ equalling 0. In our case, we calculate κ to compare the predictions of a model before and after we change the value of a factor λ across all possible setups. For example, we take the predictions from all setups in which in-context examples have the same label (the factors `one label` is present) and compare it to the case in which we have different labels for the in-context examples (the factors `one label` is absent). With all other factors being constant, we can estimate how much this factor changed the model prediction (or, inversely, how robust a model is) by calculating κ.

To ensure meaningful scores, we mask out all predictions that are not within the label distribution of the respective task. Finally, we get the overall model consistency κ_{avg}, by averaging across the κ values of all factors.

Main effects of λ Next to κ – the primary metric –, we also provide the auxiliary metric in the form of the main effects of factors. The main effects show how much the presence or absence of a factor influences the accuracy of the model on average. The main effects of the factors help to interpret their κ values: Does the change in prediction occur because the factor actually improves the model accuracy, or is it due to model inconsistency?

To obtain measures of the main effects, we fit a

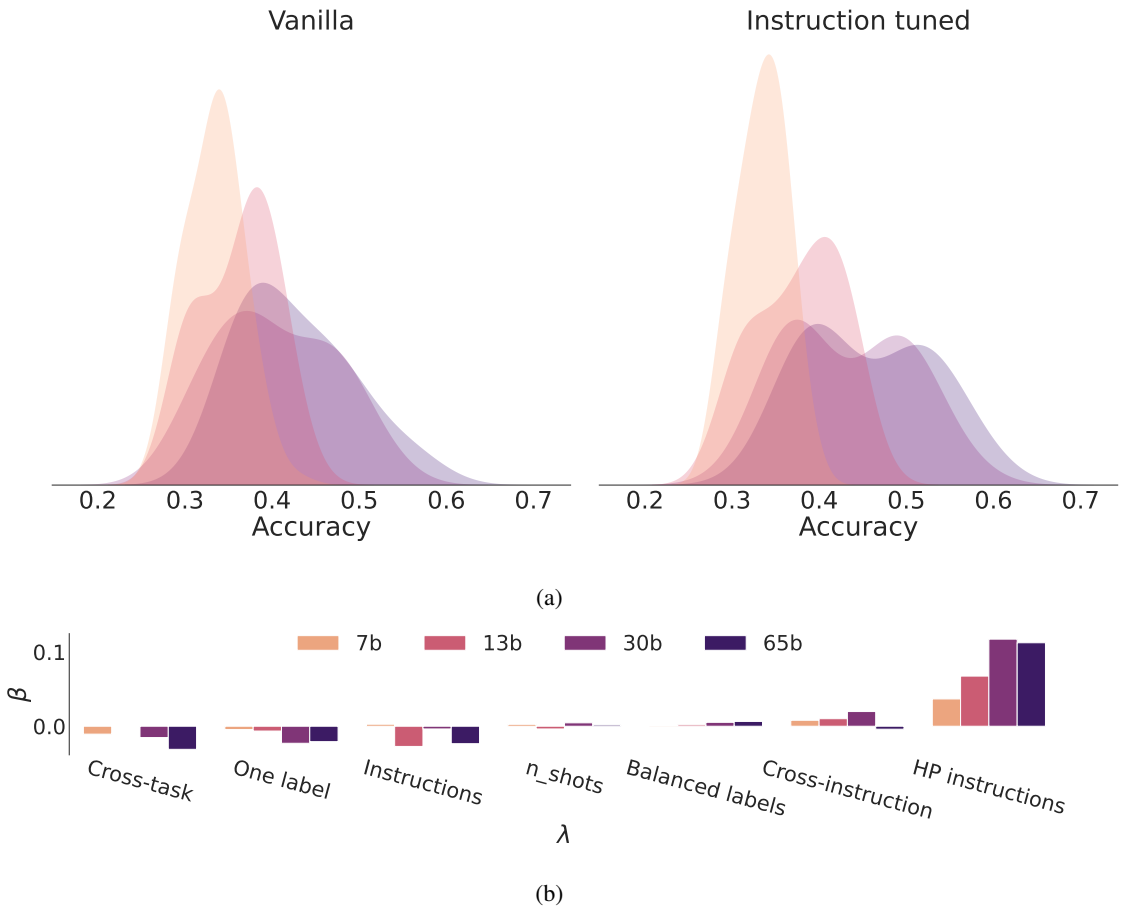


Figure 2: The following performance estimates help to put the results of our consistency metric into perspective: (a) A kernel density estimation of the accuracy scores of all eight models across all of our setups. We see how the spread of accuracy scores is high, meaning that the accuracy of a model is very different depending on the setup in which it is evaluated. (b) The β-values of the main effects of each individual factor across many different runs. The values can be directly interpreted as '*expected accuracy gain/loss*' when a factor is present compared to when it is absent.

simple linear regression model to predict accuracy scores from the presence or absence of each factor $Acc = \beta_1\lambda + \beta_0$. We can then interpret the coefficient β_1 of λ as its main effect ('How much does the factor on average change accuracy scores?').

3.4.1 Selecting instructions

To find a set of high- and low-performing instructions for the `HP instructions` factor, we run a preliminary analysis where we probe model behaviour in response to all 15 available P3 instructions for the ANLI dataset. We assess the performance of different instructions based on accuracy and consistency. A specification of the used models can be found in Table 2 and further details in Section 4.1.

We first get a general picture of each model's average consistency κ_{avg} across all instructions. We find that κ_{avg} increases with the number of parameters and is overall higher when a model has been instruction tuned (Figure 1a).

Type of learning	**Model**
ICL + vanilla	LLaMA 7B, 13B, 30B, 65B
ICL + Instruction-tuning	Alpaca 7B, 13B, 30B, 65B

Table 2: Models with their respective adaptation types, as used while selecting instructions (Section 3.4.1 and the empirical evaluation (Section 4.

We then consider the consistency of each individual instruction and find a congruent pattern of consistency across all models (Figure 1b) that corresponds generally to the accuracy scores of the same instructions (compare Figure 1c). Interestingly, we also find two groups of high-accuracy instructions making very different predictions (compare the consistency scores of 9, 10 and 15 vs. the rest). Based on these observations, we choose the two highest-

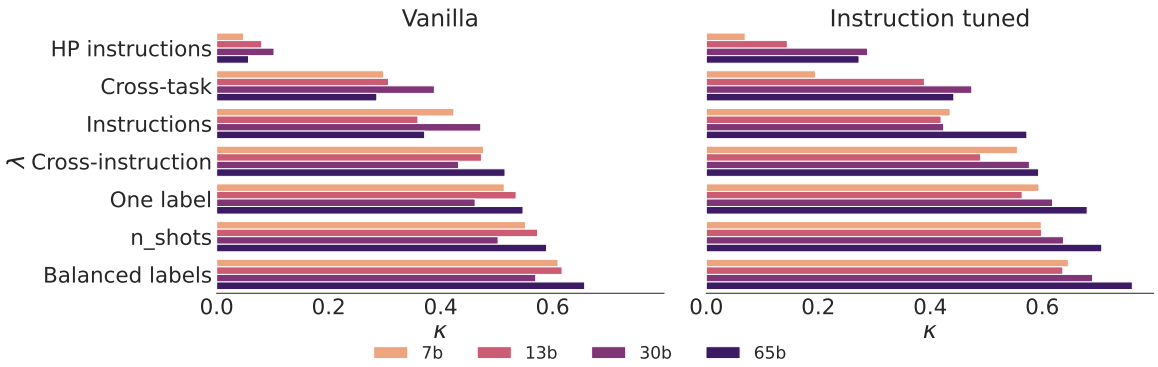


Figure 3: The consistency values comparing predictions when a specific factor λ is present or not. A value of 0 indicates a complete change of predictions while a value of 1 indicates perfect agreement. Hence, a low value indicates that a model is not robust to a change in a specific λ.

and lowest-performing instructions to constitute the `HP instructions` factor and templates 14 and 15 as realisations of the `instructions` factor. Examples of the templates that we select to construct the ICL consistency test can be found in Appendix A.

4 Empirical evaluation

In the following section, we evaluate multiple LLMs of different scales on their robustness to setup changes.

4.1 Models and sampling

We evaluate eight LLMs of different sizes and types of pretraining (for an overview, see Table 2). While 'Vanilla' models are regular pre-trained LLMs (Touvron et al., 2023), instruction-tuned models are the same models but additionally tuned to follow instructions (see e.g. Wei et al., 2022; Zhong et al., 2021) via low-rank adaptation (LoRA; Hu et al., 2022) on the alpaca self-instruct dataset (Taori et al., 2023; Wang et al., 2022). We run all models using mixed-precision decomposition as described by Dettmers et al. (2022).

To obtain a prediction from our models, we greedily sample from their probability distribution over all possible labels, with the label space being determined by the respective instruction template. However, constraining the sampling to the label distribution appears to be not strictly necessary. During our experiments, we observed that greedy sampling from the probability distribution over the whole vocabulary yields the same results.

4.2 Results

We evaluate all eight models on each setup using the full 600 data points of the ICL consistency test. We add `instruction tuning` as a custom factor to our analysis, following the description in Appendix C. For our exemplary evaluation, we focus on the ANLI dataset.

Performance distribution Figure 2a shows the distribution of accuracy scores across all runs for different models. The spread of scores is strikingly wide, with the large models scoring from below chance to up to 67% accuracy, depending on the setup. This extreme variability underlines the importance of understanding the impact of different design decisions and prediction consistency in ICL.

Main effects of λ Figure 2b presents the main effects of all factors separated by model size. The β-values show us how a change in a factor influences the accuracy of a model across all setups in general.

We can see how the choice of instructions can largely improve performance (see `HP instructions`). Surprisingly, using varied instructions for the in-context examples (see `Cross-instruction`) also slightly improves performance for 3 out of 4 models. On the other end of the spectrum, using QQP in-context (`cross-task`) as well as only providing in-context examples with only a single label (`One label`) are generally deteriorating performance, and especially so for larger models. `Instructions` shows us that choosing the correct instruction template is difficult: The average gain and loss of accuracy is ambivalent, depending on the model that is tested.

These main effects give us a general idea of the tendencies of factors and help us understand the nature of potential prediction inconsistencies in κ in the next section.

Model consistency Our primary metric, Cohen's κ, shows the consistency of a model's prediction for the same data point across many equivalent setups.

First, we ensure that our consistency is not driven by a lack of diversity in model predictions (imagine that our models always predict just a single label, as observed by e.g. Zhao et al., 2021). This is not the case (for details, see Appendix E). With this concern out of the way, we first examine separated κ for each factor λ. This measure tells us how much a model prediction changes across many setups if a specific factor is present compared to when it is absent. In Figure 3, we see kappa scores for all λs separated by model. Overall, model consistency is relatively poor: all models are susceptible to all factors, with the highest κ value scoring at 0.75.

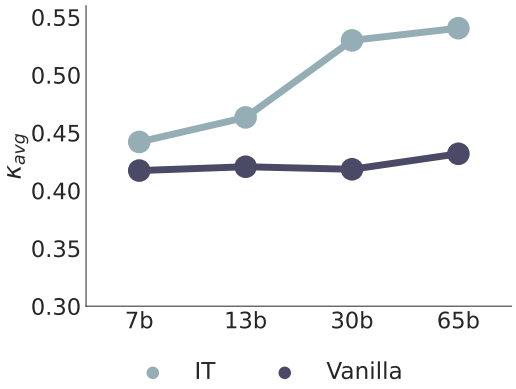


Figure 4: Average kappa per model. A high value reveals a model's robustness to setup changes.

With their predictions being influenced by factors irrelevant to the task at hand, we can conclude that the predictions of all models appear to happen on non-robust generalisations. The predictions are not reliable. Interestingly, robustness (or the lack thereof) to specific factors is highly correlated across models, with all models being more susceptible to the same factors (e.g. `HP instructions`) while relatively robust to others (e.g. `Balanced labels`).

When we accumulate κ-scores, we again observe that IT models are generally more consistent in their predictions than vanilla models (see Figure 4). Interestingly, the consistency of a model improves for IT models with increasing parameter count, while additional parameters do not improve consistency for vanilla models. This indicates that instruction tuning is especially effective in improving the robustness of larger models.

5 Conclusion

In this paper, we have introduced a new test for robustness in prompt-based learning, the ICL consistency test. The ICL consistency test evaluates the consistency of model predictions on the same data points across many different setups.

Our evaluation shows that the spread of accuracy scores across different setups is very high, indicating that model predictions are inconsistent and depend on the exact setup in which the models are evaluated. For example, the accuracy of the largest evaluated model on ANLI differs up to 40% depending on the setup in which it is evaluated. The primary metric κ of the ICL consistency test showed that the models did not perform with high consistency for any minimal setup change (i.e. across any change in factors).

The results suggest that the ICL consistency test is a good indicator of the quality of the generalisation an LLM is making: If predictions are consistent, the model correctly disregards irrelevant context information; if it is inconsistent, it lets irrelevant context information influence its predictions. The fact that state-of-the-art LLMs score low on our test highlights important room for improvement in their generalisation capacities.

Limitations

The presented ICL consistency test has several limitations in assessing model robustness. First and foremost, the test is currently only implemented for a single type of task (natural language inference), and model consistency might differ in other types of classification tasks or more open-ended answering formats such as question answering. However, we think that the performance on the ICL consistency test can be a good indicator of the quality of the generalisation that an LLM is making: If predictions are not consistent, the model is influenced by irrelevant context information, and generalisation is therefore not robust.

Another limitation of the test is that the considered factors, albeit we think our choice of factors appropriate, are in no way exhaustive and additional factors might be informative. For this reason, we added the possibility to augment the test with user-defined factors. New factors can be added to the test seamlessly.

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*.

Anas Awadalla, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt. 2022. Exploring the landscape of distributional robustness for question answering models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5971–5987, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Stephen Bach, Victor Sanh, Zheng Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-david, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Fries, Maged Alshaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-jian Jiang, and Alexander Rush. 2022. PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104, Dublin, Ireland. Association for Computational Linguistics.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *ArXiv preprint*, abs/2302.04023.

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*.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *ArXiv preprint*, abs/2208.07339.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

D. Hupkes, M. Giulianelli, V. Dankers, et al. 2023. A taxonomy and review of generalization research in nlp. *Nature Machine Intelligence*, 5:1161–1174.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *ArXiv preprint*, abs/2211.09110.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. 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? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022. Reframing instructional prompts to GPTk's language. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 589–612, Dublin, Ireland. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Xenia Ohmer, Elia Bruni, and Dieuwke Hupkes. 2023. Evaluating task understanding through multilingual consistency: A chatgpt case study. *ArXiv preprint*, abs/2305.11662.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. 2023. Prompting gpt-3 to be reliable.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *ArXiv preprint*, abs/2302.13971.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions. *ArXiv preprint*, abs/2212.10560.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 4144–4150. ijcai.org.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. 2023. Larger language models do in-context learning differently. *ArXiv preprint*, abs/2303.03846.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Kang Min Yoo, Junyeob Kim, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang-goo Lee, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2422–2437, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2856–2878, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Prompt template examples

In the following, we provide more details on the instruction templates (Bach et al., 2022) used to construct the ICL consistency test.

A.1 P3 details – names

Names of all available P3-instructions, ordered as in Figure 1

1. 'MNLI Crowdsource'
2. 'Guaranteed Possible Impossible'
3. 'Always Sometimes Never'
4. 'Consider Always Sometimes Never'
5. 'Does This Imply'
6. 'Guaranteed True'
7. 'GPT 3 Style'
8. 'Take the Following as Truth'
9. 'Must Be True'
10. 'Based on the Previous Passage'
11. 'Should Assume'
12. 'Can We Infer'
13. 'Justified in Saying'
14. 'Does It Follow That'
15. 'Claim True False Inconclusive'

A.2 P3 details – examples

High-performing templates 'Claim true false inconclusive'

[...]

Jonathan Smith (born January 17, 1971), better known by his stage name Lil Jon, is an American rapper, record producer, and DJ. He was the frontman of the group Lil Jon & The East Side Boyz, which he formed in 1997, and they released several albums until 2004. Based on that information, is the claim: "Jonathan Smith spent much of his time in China." true, false, or inconclusive?

ANSWER:

High-performing templates 'Does it follow that'

[...]

Given that Jonathan Smith (born January 17, 1971), better known by his stage name Lil Jon, is an American rapper, record producer, and DJ. He was the frontman of the group Lil Jon & The East Side Boyz, which he formed in 1997, and they released several albums until 2004. Does it follow that Jonathan Smith spent much of his time in China. Yes, no, or maybe?

ANSWER:

Low-performing templates 'MNLI crowdsource'

[...]

Jonathan Smith (born January 17, 1971), better known by his stage name Lil Jon, is an American rapper, record producer, and DJ. He was the frontman of the group Lil Jon & The East Side Boyz, which he formed in 1997, and they released several albums until 2004. Using only the above description and what you know about the world, "Jonathan Smith spent much of his time in China." is definitely correct, incorrect, or inconclusive?
ANSWER:

Low-performing templates 'Guaranteed possible impossible'

[...]

Assume it is true that Jonathan Smith (born January 17, 1971), better known by his stage name Lil Jon, is an American rapper, record producer, and DJ. He was the frontman of the group Lil Jon & The East Side Boyz, which he formed in 1997, and they released several albums until 2004.

Therefore, "Jonathan Smith spent much of his time in China." is guaranteed, possible, or impossible?

ANSWER:

B Factor details

In the following, we provide a more detailed description of the factors shortly summarized in Section 3.3 and also offer our motivation to include these factors.

n-shots The number of examples in-context is a factor that is always present in any in-context learning setting. As such, it is an essential factor to include, as it has high practical significance and should be considered with caution if it results in low κ values. We introduce few ($k = 2$) and many ($k = 5$) examples as a factor.

HP instructions We have seen how some instructions produce consistent and relatively well-performing responses across different models while others do not (see Section 3.4.1. We add this factor to increase the difficulty of the task: A robust model should be able to predict a consistent label in response to either of these equivalent instructions. We chose the two best and two worst-performing templates[2] from our analysis in Section 3.4.1.

Balanced labels Zhao et al. (2021) showed how a majority label in-context can influence the distribution of model outputs. We, therefore, compare contexts with balanced in-context label distribution with randomly sampled labels.

Cross-instruction We include cross-templates as a factor to assess model robustness to shifts in the label space $\mathcal{C}$ and surface form of instruction formulation. Previous research has shown how in-context learners are sensitive to the instructions (Mishra et al., 2022) as well as the label distribution $\mathcal{C}$ (Min et al., 2022). The experiments of Min et al. (2022) represent an extreme case in which $\mathcal{C}$ is resampled as random tokens. While these edge cases are theoretically attractive, we here change this scenario to a more common one, where instructions and labels are semantically equivalent but have different surface forms. We randomly sample from the set of available p3 instructions to obtain the in-context examples. Surprisingly, when testing this factor, we find that almost all models are robust to semantic-invariant changes to the in-context instructions (see Figure 5). This happens despite changes in the label space and substantial changes in surface form and format across the used instructions.

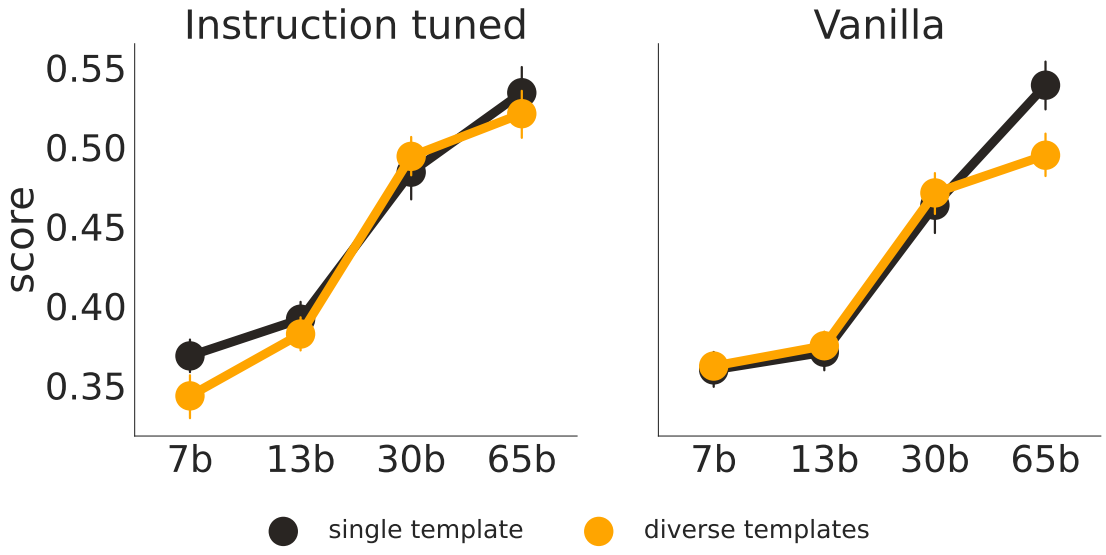


Figure 5

Cross-task In `cross-task`, we exchange the task of the in-context examples such that the only consistency between in-context and target examples is the general format (x followed by y) and the truthfulness

[2]See Appendix A for an example of the instructions

of the x to y mapping. To see whether conditioning on a fixed label space matters, we add tasks with a discriminative (QQP) and a generative (SQuAD; Rajpurkar et al., 2016) objective as different factors. Compared to a zero-shot baseline, we can see that especially large models can benefit from conditioning on other tasks (Figure 6).

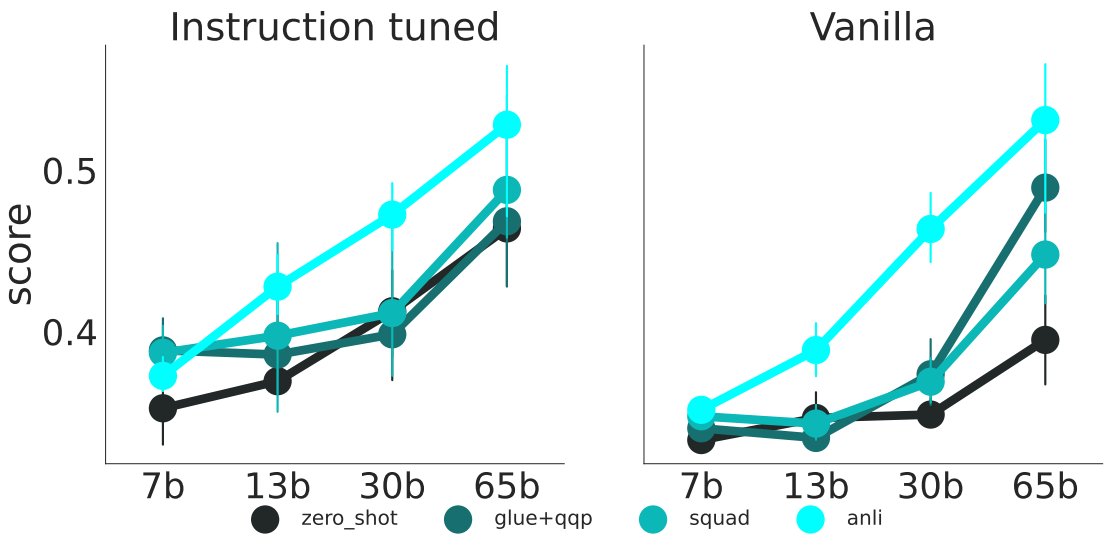


Figure 6

For our test, we only include QQP as an in-context task, as SQuAD is incompatible with many other factors (such as `balanced labels`, `one label` aso...)

Instructions Besides the performance of the instructions, we are also interested in how consistent model behaviour is across instructions with similar performance. We bin the two high-performing instructions into a new factor to get insights into this.

C Introducing custom factors

The ICL consistency test allows the addition of additional user-defined factors. This is useful if factors should be evaluated that are related to modifications of the model (e.g. whether it was instruction-tuned Wei et al., 2022, or not) or when the model was evaluated in a different way (e.g. whether we calibrate our output probabilities Zhao et al., 2021, or not). Note that adding a factor in this way will change the overall results of the analysis (see Section 3.4 for more details). Alternatively, the task can be evaluated separately for either user-defined factors.

D Details setup_IDs

Setups are defined by the presence and absence of factors. The setup_IDs reflect how the factors make up the setup (see Figure 7). Setup_IDs can also be converted into a human-readable format by using the '`_convert_numeric_id_to_dict`'-method of the '`task`'-object in the code submission to the GenBench CBT.

E Prediction diversity

Models could achieve a high consistency score by always predicting the same label. We check whether models tend to do so by calculating the entropy of a model's predictions across all data points in the ICL consistency test. This allows us to estimate whether a model is biased toward predicting a single label (low entropy). An unbiased model's prediction should be close to the entropy of the target distribution $\mathcal{H}(Y)$. We find that smaller models have a larger bias towards predicting a single label (lower prediction entropy),

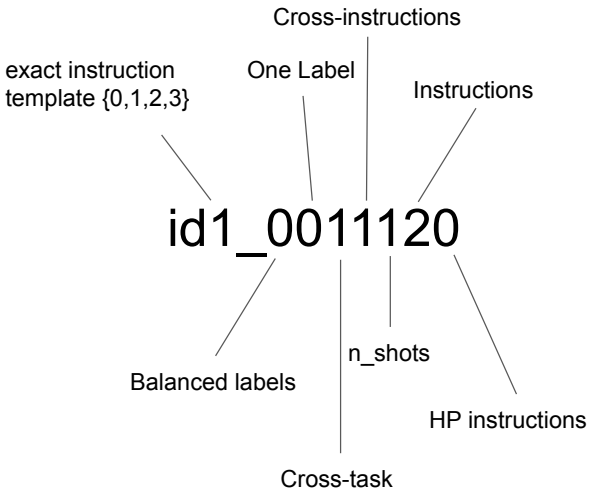


Figure 7: An illustration of the meaning of every digit in a setup_ID: A factor is either absent (0), present (1) or irrelevant for the setup (2).

while larger and IT models get closer to $\mathcal{H}(Y)$ (see Figure 8). This hints that we might overestimate the κ values for smaller vanilla models.

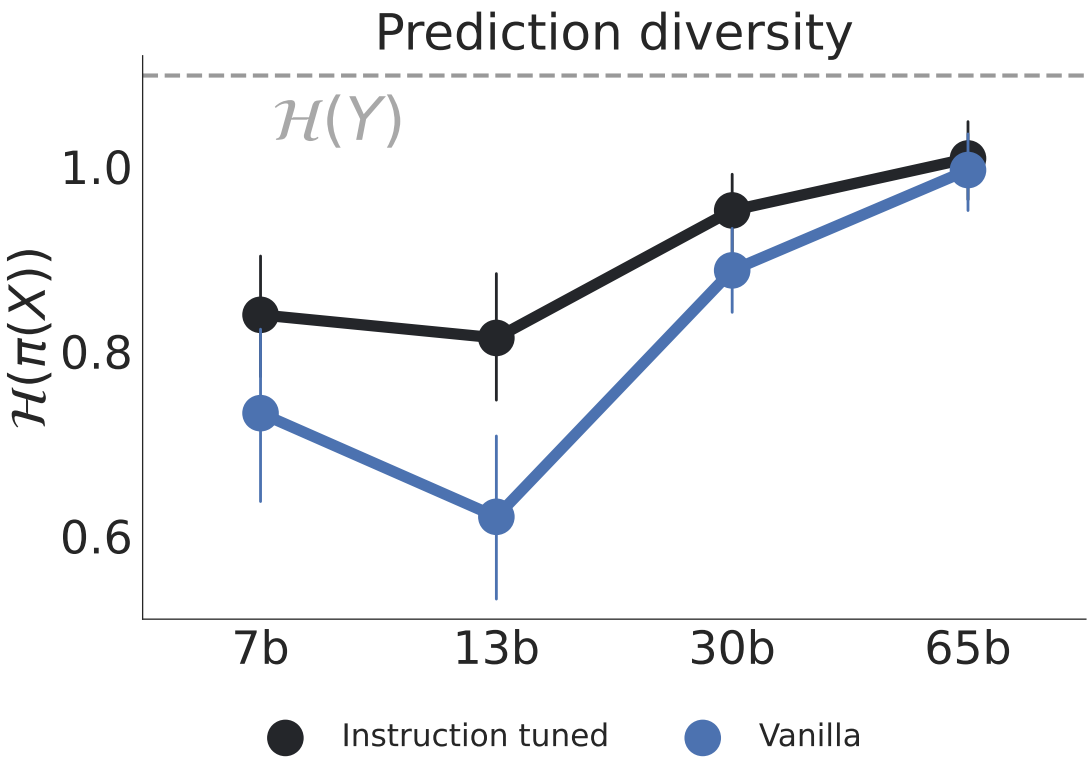


Figure 8: The diversity of model predictions across the whole dataset measured through entropy. A value close to 0 indicates that a model is strongly biased to always predict the same label. A value close to the entropy of the true label distribution $\mathcal{H}(Y)$ indicates that the model has no bias towards any specific label.