arXiv:2405.14782v1 [cs.CL] 23 May 2024

Lessons from the Trenches on Reproducible Evaluation of Language Models

Stella Biderman[1*], Hailey Schoelkopf[1*], Lintang Sutawika[1*],
Leo Gao[1], Jonathan Tow[2], Baber Abbasi[1], Alham Fikri Aji[3], Pawan Sasanka Ammanamanchi[4], Sidney Black[1], Jordan Clive[5], Anthony DiPofi[1], Julen Etxaniz[6], Benjamin Fattori[1], Jessica Zosa Forde[7], Charles Foster[8], Mimansa Jaiswal[9], Wilson Y. Lee[10], Haonan Li[3,11], Charles Lovering[12], Niklas Muennighoff[13], Ellie Pavlick[6], Jason Phang[1,14], Aviya Skowron[1], Samson Tan[15], Xiangru Tang[16], Kevin A. Wang[6], Genta Indra Winata[17], François Yvon[18], and Andy Zou[19]

[1]EleutherAI, [2]Stability AI, [3]MBZUAI, [4]IIIT Hyderabad, [5]Chattermill AI, [6]HiTZ Center - Ixa, UPV/EHU, [7]Brown University, [8]Finetune, [9]University of Michigan, [10]HubSpot, [11]LibrAI, [12]Kensho, [13]Contextual AI, [14]New York University, [15]Amazon, [16]Yale University, [17]HKUST, [18]Sorbonne University, [19]CMU

[*]Equal Contribution

Abstract

Effective evaluation of language models remains an open challenge in NLP. Researchers and engineers face methodological issues such as the sensitivity of models to evaluation setup, difficulty of proper comparisons across methods, and the lack of reproducibility and transparency. In this paper we draw on three years of experience in evaluating large language models to provide guidance and lessons for researchers. First, we provide an overview of common challenges faced in language model evaluation. Second, we delineate best practices for addressing or lessening the impact of these challenges on research. Third, we present the `Language Model Evaluation Harness` (`lm-eval`): an open source library for independent, reproducible, and extensible evaluation of language models that seeks to address these issues. We describe the features of the library as well as case studies in which the library has been used to alleviate these methodological concerns.

1 Introduction

Evaluation on shared benchmark tasks is a crucial tool used to track and communicate progress in the machine learning and language modeling communities (Ruder, 2021). Benchmarks are used to track progress toward shared community goals and to demonstrate the improvements of newly proposed methods over prior baselines. Evaluation practices thus play a crucial role in the direction of the field: inconsistencies or biases in evaluation practices can lead to skewed performance comparisons, which may influence the direction of future research and the adoption of new methods by the community (Dehghani et al., 2021) or lead to adverse effects from deploying suboptimal or harmful models (Bender & Friedman, 2018) on tasks for which they are ill-suited (Raji et al., 2022).

Unfortunately, transparent and reproducible evaluation of large language models is very challenging. In our research we have frequently struggled to reproduce the results reported in various papers as well as carry out new evaluations ourselves. To address this problem we built the Language Model Evaluation Harness (Gao et al., 2021), a flexible evaluation library that serves as **research infrastructure for evaluation**. Our goal with `lm-eval` is to

enable researchers to run any benchmark on any model as easily as possible, while also making it easy for creators of new model inference libraries or evaluation benchmarks to connect their work to the broader ecosystem.

Over the past three years, the design of `lm-eval` has evolved as the needs of the open source community and our understanding of best practices for language model evaluation have evolved. In this paper we detail lessons learned that have been especially beneficial to obtaining useful and rigorous findings. We highlight several commonly-faced challenges in evaluating language models, including the difficulty of assessing the correctness of natural language responses, challenges in benchmark design, and the dependence upon implementation details that are often obscured or unreported (Section 2). We then discuss best practices we've identified to improve how to communicate results and improve evaluation rigor in the language modeling community, despite these challenges (Section 3). Finally, we detail how we have used our learnings to inform the design of `lm-eval` (Section 4).

2 Challenges in Evaluating Language Models

2.1 Evaluating and Scoring Natural Language Abilities

The biggest challenge in language model evaluation is a concept we term **the Key Problem**: When evaluating language models, there can be many semantically equivalent but syntactically different ways of expressing the same idea. In an ideal world, we would have a way to automatically detect when two sentences express the same content but in different words. Unfortunately, our best tools for determining whether two sentences are semantically equivalent *are the very models we are seeking to evaluate*. This problem drives many of the approaches to LM benchmarking, and many problems in LM evaluation stem from there not being any silver bullets for solving the Key Problem.

In principle, this would be solvable by simply having expert human annotators score model responses for correctness. The main reason this is not ubiquitous is cost: performing accurate human studies is not only difficult and time-consuming but also very expensive due to fair compensation, pricing smaller actors or organizations out of performing such evaluations. Additionally, there are other reasons relying on solely human assessments must be done with caution: they can be flawed and biased, especially for complex judgments such as factuality (Hosking et al., 2024; Xu et al., 2023; Wu & Aji, 2023). Expert, trained human judgment can alleviate these issues but is inherently non-scalable.

To address the high costs of manual human evaluation, automated metrics are often used. These offer notable advantages in that they are (theoretically) fully reproducible, far easier and cheaper to compute, and can avoid some of the issues faced by human studies (Wei & Jia, 2021; Freitag et al., 2021; Amidei et al., 2020). Automated metrics such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) seek to directly solve the Key Problem by measuring the distance from a generated response to a gold-standard one, such as via counting the n-gram overlap between the two texts. Heuristic-based metrics such as BLEU (and its derivatives) have flaws (Callison-Burch et al., 2006) and present reproducibility challenges (Marie et al., 2021), but can be useful. More recently, model-based metrics have recently gained momentum through evaluation methods that leverage large language models as a grader (Kim et al., 2024; Wang et al., 2024; Liu et al., 2023b), especially as proxies for human preference evaluation (Zheng et al., 2023), but these are known to be flawed (Wang et al., 2023; Shen et al., 2023a; Zeng et al., 2024; Hu et al., 2024; Liu et al., 2023c; Chen et al., 2024) and suffer from similar reproducibility issues as BLEU, ROUGE, and their variants.

The Key Problem can alternately be sidestepped by artificially restricting the answer space. The most prevalent way to achieve this is to reframe questions as multiple choice problems, with a single gold target answer and a finite, static set of possible responses (Hendrycks et al., 2020; Srivastava et al., 2022; Li'evin et al., 2022; Lin et al., 2022; Robinson et al., 2023; Holtzman et al., 2022). Alternatively, when a reference answer is known, one can perform string-matching approaches heuristically to determine whether the model's answer matches the ground truth (Dua et al., 2019; Joshi et al., 2017; Hendrycks et al., 2021).

This challenge does not necessarily impact other applications of language models and related technologies, such as playing games where it easy to check that the game has ended (Romstad et al., 2008; Silver et al., 2018; , FAIR), more constrained scientific applications (Jumper et al., 2021; Ahdritz et al., 2022), or domains where we have *practically usable verifiers* even when the solutions are not checkable in all contexts (Biderman, 2020; Biderman & Raff, 2022; Lewkowycz et al., 2022). In the case of LLMs, the most notable cases where this ground-truth verifier is known are coding and mathematics problems, although the verifiers used, such as unit tests, may still break down in edge cases (Liu et al., 2023a)

2.2 Benchmark Design and Validity

Typically, we do not care about the actual numeric score of a model on a benchmark. Instead, we desire the benchmark to be a useful proxy for some real-world phenomenon. The *validity* of an evaluation is the extent to which these correlate (Messick, 1994). For a recent overview of validity concerns in NLP benchmarking, see Subramonian et al. (2023). Also see Raji et al. (2021); Saphra et al. (2023); Davis (2023) for extended discussion of construct validity in LLM evaluation.

While validity is an ongoing problem in language model evaluation, we focus on mitigating other concerns first: as we will describe, `lm-eval` is designed to ensure measurements are *consistent* across runs and models, regardless of (construct) validity. This is due to our goal of building *research infrastructure for evaluations*. While we as researchers prefer some evaluation benchmarks to others, our goal is to enable researchers to run any evaluation benchmarks on any models.

2.3 Implementation Difficulties and (Ir)Reproducibility

Once a benchmark has been designed, it then needs to be implemented by machine learning researchers around the world to see use in driving progress in the field. This introduces a host of new challenges that need to be addressed in order to ensure that everyone is evaluating models on a benchmark in the same fashion when comparing results. This adaptation process can introduce inconsistencies and make it difficult to draw conclusions across different implementations. Researchers must *adapt* it to their own workflows and libraries for the purposes of actually adopting the benchmark in their research.

2.3.1 "Minor" Implementation Details Matter

The importance of interoperability and full reproducibility stems from the fact that language models are incredibly sensitive to precise details that may not be obvious to practitioners. Even minor variations in prompts, formatting, or other implementation details can significantly impact the performance and validity of evaluations (Weber et al., 2023; Sclar et al., 2023; Mizrahi et al., 2024; Alzahrani et al., 2024; Lu et al., 2022; Webson & Pavlick, 2022; Min et al., 2022). Without access to the original evaluation code, when re-implementing evaluation procedures from scratch is required, it is nearly impossible to account for all the subtle details that can affect outcomes. As a result, these implementations are likely to diverge in ways that make it extremely difficult to ensure fair comparisons across works, even when evaluating on the same benchmark. Even having the prompts reported in a paper is no substitute for having access to the actual evaluation code: prompts in papers are often incorrect or difficult to map to the exact code implementation because they've been stylized to be human-readable.

2.3.2 Lack of Agreement About "Apples to Apples"

Even assuming that benchmarks are implemented consistently across works, the question of *how to draw fair comparisons* across models and methods is still difficult for LMs.

For instance, different instruction-tuned models may be trained to expect certain formats (Taori et al., 2023; Sanh et al., 2022; Wei et al., 2022) – using these models' intended prompt formats can make the evaluation tasks inherently different or change their difficulty, but not using these can also bias against models trained with formats not matching tasks'

"standard" prompting styles. Likewise, if an original benchmark implementation (including prompting and postprocessing) is tailored for a specific model, other models trained differently will suffer, artificially skewing perceptions of what techniques are effective.

Likewise, some questions of how to set up controlled experiments are still open–is it ideal to normalize performance and comparisons by the number of parameters? Training FLOPs? Inference cost? Must training data be held equal? How should models which can leverage external resources such as retrieved documents or external tools be compared? These questions are all context-dependent but can impact findings significantly. For example, Wang et al. (2022) explore comparisons across architectures and training objectives, and choose to normalize for FLOPs, thus comparing encoder-decoder models with double the parameters to decoder-only models. Comparing results of models with equivalent training FLOPs, regardless of the allocation of those FLOPs, is commonplace (Hoffmann et al. (2022); Peng et al. (2023); Touvron et al. (2023), *inter alia*). However, in a more memory-constrained setting, comparing models equi-parameter may be more logical. While this is not inherently problematic, as different application contexts motivate different evaluation criteria, it is common to gloss headline claims as referring to the general case without paying significant attention to such caveats.

2.3.3 Comparisons with Prior Work are Expensive (and Sometimes Impossible)

Setting aside the question of establishing fair comparisons between methods or models, an additional key challenge in language modeling research is that many barriers prevent thorough comparison with related work.

Many LMs developed by industrial labs, often used as reference points for benchmarks, have never been released externally (Chowdhery et al., 2023; Hoffmann et al., 2022), preventing comparisons except by pulling unverified evaluation numbers from technical reports. Those models that have been made available via APIs may non-transparently not match the published versions or otherwise be modified for deployment. Additionally, these API models are quickly *deprecated* and no longer accessible, rendering slews of work no longer reproducible[1]. API access, especially for large volumes of evaluation, is quite expensive.

Further, a growing number of companies no longer make *base* language models available, but enforce interaction via a chat interface or API, which may include product features such as personalization[2], safety controls[3] or domain-specific tooling[4]. Attempts to compare these closed *systems*, which integrate a language model along with proprietary features, introduce a whole new set of complications.

2.4 Fast-changing Progress and Conventions

Due to the time-consuming nature of developing good benchmarks and the rapid pace of change in NLP research in the past decade, many widely used language model evaluation benchmarks do not represent the current paradigm of how language models are trained. This has two major impacts:

1. Benchmarks are being used for purposes they were not originally designed for or designed for validity under: for example, a large number of benchmarks have been built around fine-tuning on a known training set and closed space of labels (Wang et al., 2019b;a).

2. There is no "ground-truth" implementation from the original benchmark authors for many of these popular benchmarks "retrofitted" to be used with autoregressive LMs. In the absence of a clear standard, the community's methodology for evaluating on

---

[1]Notably, OpenAI's `code-davinci-002` model was deprecated in January 2024, making at minimum hundreds of research studies irreproducible.

[2]`https://blog.google/technology/ai/bard-google-ai-search-updates/`

[3]`https://openai.com/blog/our-approach-to-ai-safety`

[4] ChatGPT vs. Microsoft Copilot: What's the difference?

these benchmarks may be fragmented or undocumented (Clark et al., 2018; Paperno et al., 2016).

To illustrate the effects of this development timeline, Figure 1 shows how many prominent LM benchmarks were designed prior to shifts such as in-context learning and chat interaction, and therefore were not designed to take these formats and approaches into account. This can affect validity or difficulty in unforeseen ways.
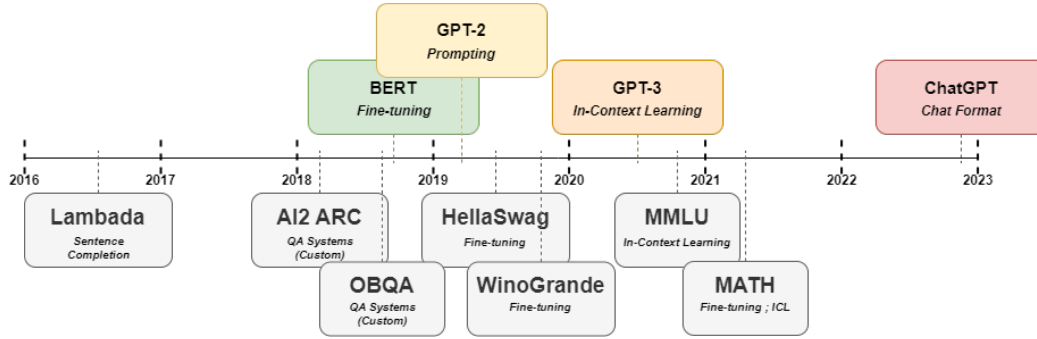

Figure 1: A timeline showing the relative release dates of a selection of notable benchmarks used to evaluate LMs, as compared to the release dates of BERT (Devlin et al., 2018), GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), and ChatGPT, used as approximate stand-ins for shifts in how the community uses and therefore evaluates LMs. Common practice for evaluating autoregressive language models today diverges from the method described in the paper for all listed tasks except MMLU and MATH.

3 Best Practices for Language Model Evaluation

While LM evaluation is difficult and suffers from a number of challenges as we have described, there are measures that can be taken to significantly improve current practices. We provide our high-level recommendations regarding such measures, and detail our motivations briefly for each.

Always share your exact prompts and code

- If possible, full *evaluation code* including the full prompts used should also be provided for reproducible evaluation runs, as well as further identifiers such as links to specific commits used. Failing this, sharing prompts is often not done, but can drastically improve reproducibility.

- For fair comparison against other models, evaluation should be done with the same set of prompts unless there's a good reason not to. **Prompts should not be optimized for performance on a given model but not others, and the amount of prompt engineering done should be disclosed.**

Avoid copying results from other implementations

- Comparing results across papers can be misleading due to a wide range of experimental differences, including prompts, sample size, metric calculation, and more (Marie et al., 2021).

- Results should **not** be copied or reported from other papers (Marie, 2022) whenever possible, unless one can verify that the exact same code has been used to run the experiments in those papers. If such copying is unavoidable, it should be clearly marked as such and treated carefully.

Always provide model outputs

- Providing model outputs alongside evaluation code can allow others to recalculate scores based on these artifacts, which can be useful for performing statistical significance testing and for assessing the impact of different evaluation metrics or scoring approaches.

- Evaluation of large models or APIs can be quite costly–sharing such artifacts allows researchers without access to significant compute to participate in evaluation research.

- Finally, sharing outputs can allow results on API models to be reproduced to some extent, even if the models are subsequently deprecated.

Perform qualitative analyses

- Qualitatively review a small batch of results and outputs before testing at scale: it is very easy to have bugs in your generation code, especially when working with multiple sets of benchmarks, prompts, and models of different architectures. Catching issues early can save a lot of time and compute, and increase confidence in results.

- Quantitative scores only provide so much information. To understand why a model is scoring so well or so poorly, it is important to do some sort of qualitative error analysis. This can sometimes reveal superficial errors that are easier to correct with post-processing (Bawden & Yvon, 2023), or more fundamental errors.

Measure and Report Uncertainty

- Most works on language modeling do not perform statistical significance testing, which can significantly decrease confidence in results (Marie et al., 2021).

- Although costly, reporting results run over more than one random seed can dramatically boost the validity and utility of results. For example, averaging across model runs (Sellam et al., 2022), or averaging over multiple selections of few-shot examples.

- Even when not retraining models, statistical analysis can bound the expected variation across model training runs (Jordan, 2023).

4 The Language Model Evaluation Harness

Informed by these practices we have built `lm-eval`. Unlike subsequent[5] work on unified benchmarking libraries (Liang et al., 2023; Srivastava et al., 2022; Barton, 2024), the Evaluation Harness does not seek to solely prescribe what the correct benchmark or evaluation protocols to use are, and allows users to select their desired tasks and use cases.

The role of the `lm-eval` is to solve the *orchestration problem*: previously, performing thorough LM evaluations would require painstaking re-implementation of previous tasks (likely to introduce subtle methodological divergences) or individually installing, debugging, and using dozens of small libraries. Our goal is to make it easy to allow researchers or library users to simply install one codebase, and run their method plus selected baselines on their desired tasks in a controlled fashion. At the same time, we strive to design best practices into the functionality of the library itself, so that the default and easy-to-use functionality guides users to follow best practices.

---

[5] `lm-eval` was originally built in 2021 and has been in continuous use at EleutherAI and elsewhere since then, despite not being formally introduced in any papers.

4.1 Design

We provide an overview of `lm-eval`'s major components and design philosophy. At its core, `lm-eval` allows for the contribution of two types of implementations: evaluation **Tasks** and integrations with novel **LM** implementations.

Tasks `lm-eval` is built around modular implementations of evaluation tasks, implemented as a **Task** class using a common API. This allows tasks to be collected in a common library, for new tasks to be extended or implemented easily, and for novel tasks to be easily shared reproducibly among practitioners or other library users. Users can implement tasks either via YAML-based configuration files or via subclassing the provided **Task** class and providing custom code for specific methods. In Figure 2, we show an example of the evaluation logic packaged within a **Task** class.
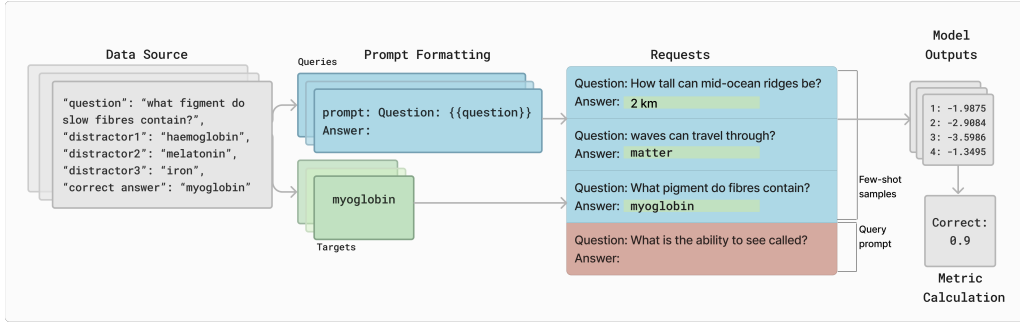

Figure 2: The operations performed by a **Task** object in `lm-eval`. Tasks are configured by YAML files or as a Python subclass, and encompass 1) a data source (using the `Datasets` library(Lhoest et al., 2021)), 2) tools for defining prompts and format, 3) mapping these prompts to rendered inputs and expected output type from an **LM** in the form of **Requests**, and 4) rules for post-processing the **LM**'s outputs and calculating the final task metrics.

We provide a number of implementations for common tasks, and accept new tasks sourced from the community. We strive to match the paper originally introducing a benchmark dataset in its methodology, including using the same prompts if applicable. For tasks such as those introduced prior to prompted evaluation becoming the standard, we source evaluation methodology from the paper first posing the evaluation dataset as a prompted task. For example, we implement many tasks as adapted for in-context learning by Brown et al. (2020).

LMs The next core piece of `lm-eval` is the **LM** API. Because effective *orchestration* is our core goal, we allow arbitrary software libraries or (autoregressive) language model architectures to extend a provided interface for **LM** objects.

For ease of use, and compartmentalization of the model definition and external library integrations for custom models away from core evaluation logic, we assume that LMs operate upon dispatched **Requests** which consist of mapping *string inputs* to some *string or probability* as output. We thus abstract tokenizers away within the **LM** class, and treat a neural language model combined with its tokenizer as a single system being evaluated.

LMs implement a simple interface, consisting of several types of **Requests** in order to be used within the library for all supported tasks.

Request Types We allow for 3 core types of **Requests** that may be sent to a language model, which consist of distinct types of *measurements* that can be performed to observe a model's response or latent capabilities in a prompted format. These are:

- (Conditional) Loglikelihoods (`loglikelihood, multiple_choice`) - computing the probability of given output string(s), conditioned on some provided input.

- Perplexities (`loglikelihood_rolling`) - measuring the average loglikelihood or probability of producing the tokens in a given dataset.
- Generation (`generate_until`) - generating text until a given stopping condition is reached, from a model conditioned on some provided input.
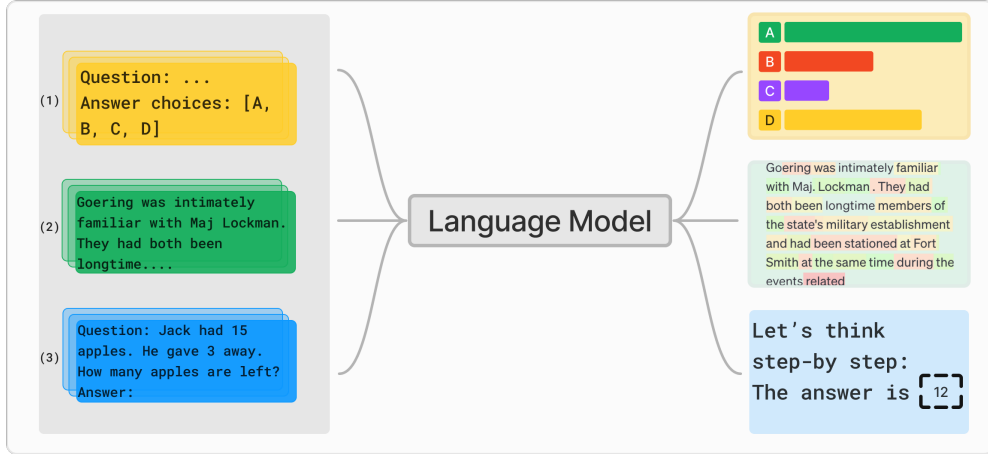

Figure 3: Overview of the three core **Request** types supported by our evaluation framework. These include (1) conditional loglikelihoods, (2) perplexities, and (3) generation-based requests.

Provided with these three primitive operations, we are able to implement the major ways in the literature that have been used to evaluate LMs (Gao et al. (2020), Brown et al. (2020), *inter alia*). While these high-level approaches are standard, they all contain a number of subtle implementation decisions which are often not disclosed in papers. Therefore, we include a full formal description of common implementation details involved in ours and others' approaches within Appendix A for completeness, which we hope will be a useful contribution to the literature.

4.2 Addressing Challenges and Incorporating Best Practices

Here we detail how we position `lm-eval` to address the issues mentioned in Section 2 and incorporate the recommendations in Section 3, in order to encourage a more robust evaluation ecosystem.

Reproducibility `lm-eval` encourages and enables reproducible evaluation in several ways. First, by providing a standardized implementation of many common tasks, practitioners can report on these tasks and ensure they are evaluating on the same prompt and implementation as other users of the library.

Alongside task results we report a `version` field, incremented each time a task must be modified in a way that affects its scoring. Therefore, in the case where task implementations have bugs or must otherwise be updated, one can still reference the version of the task used, to ensure future research can reproduce reported results.

While this is not a panacea for the costs of comparing to prior work, and rerunning baselines oneself is advised, when prior work uses our library one can be confident that the results from prior work match what one would have gotten had one rerun it oneself using that version of the library (Beeching et al., 2023).

Qualitative Analysis `lm-eval` provides support for performing qualitative analysis of evaluation scores. In keeping with our recommended best practices, we implement the

following, which allow for qualitative checks to be a core part of the evaluation workflow when using `lm-eval`:

- We allow for artificially `limiting` the amount of samples used for a given evaluation run, to enable code to be tested and outputs to be reviewed in small batches prior to full evaluation runs.
- Per-sample logging is supported, for post-hoc reproduction of scores or error analysis of model mistakes or evaluation implementation.

Statistical Testing `lm-eval` reports the standard error (SE) of most supported metrics[6], calculated by either bootstrapping or dividing the sample standard deviation by the root of the sample size.

By reporting these SE calculations prominently in every evaluation run, we make it easy for practitioners to add simple statistical measures such as confidence intervals to their results. While we believe more rigorous and widespread statistical testing in LM evaluation is still needed, we hope that this will spur the community to report and be more aware of statistical significance concerns and lower the difficulty of reporting such measures.

Note that the standard errors described here refer to an estimate of the variance in the observed scores should the evaluation data be recollected from the same distribution (Recht et al., 2019; Zhang et al., 2024). Other forms of variance, such as temporal drift (Longpre et al., 2023), retraining models with different seeds (a paradigm we are currently experimenting with), or variance across prompts (Sanh et al., 2022; Muennighoff et al., 2022) must be calculated differently.

5 Case Studies

Finally, we demonstrate `lm-eval`'s utility for improving evaluation rigor and understanding via case studies of its successful usage. Additional case studies can be found in Appendix B.

5.1 Multiprompt Evaluation with the BigScience Workshop

Even when scoring criteria are held identical, the specific prompt of an evalution task can heavily impact results. In collaboration with the BigScience Workshop, the PromptSource (Bach et al., 2022) library was added to a fork of `lm-eval` to enable easy evaluation across many different prompt templates[7] for the first time. Most papers that came out of the BigScience Workshop used this functionality to report distributions of scores across different prompting set-ups (Sanh et al., 2022; Muennighoff et al., 2022; Yong et al., 2023; Workshop et al., 2023). PromptSource, along with other innovations introduced in the BigScience fork, is now supported natively in `lm-eval`. We have also further extended our functionality to enable people to use the Jinja templating language directly in their configuration files to make it easy to define custom evaluation templates by hand and algorithmically, regardless of whether they use prompts from the PromptSource library.

While the approach used by BigScience in reporting distributions across prompts has not been widely adopted, the idea that prompts should be considered as part of the evaluation set-up has become widely accepted. We hope that future research will especially continue to focus on collecting *realistic prompts* (Shen et al., 2023b; Xie et al., 2023; Hofmann et al., 2024) or measuring the extent to which results with particular common set-ups generalize to more realistic use-cases (Lyu et al., 2024), or otherwise investigate prompting as a human-computer interaction problem.

---

[6]The standard error is always the standard error of the evaluation metric. In most, but not all, cases this is the standard error of the mean, as most evaluation benchmarks report the mean score as their final metric.

[7]https://github.com/bigscience-workshop/lm-evaluation-harness
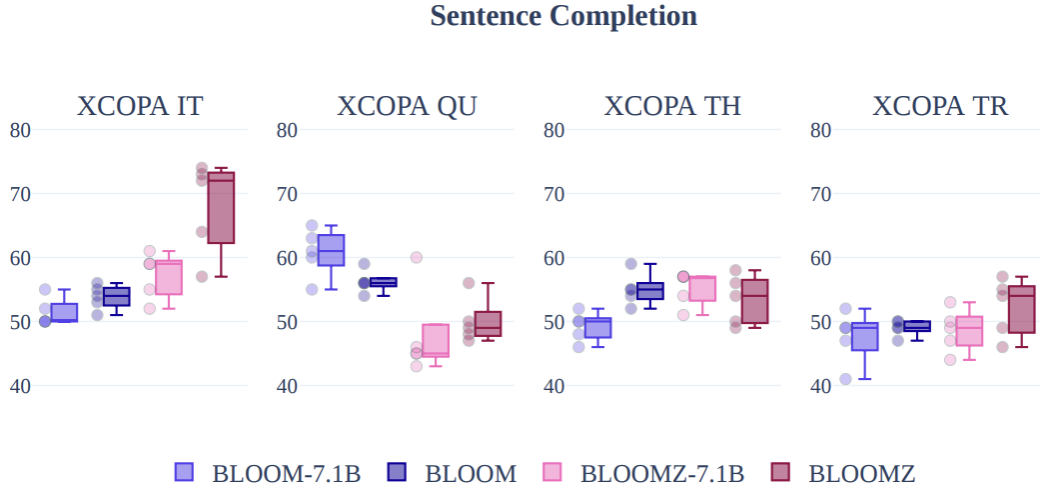
Sentence Completion

Figure 4: An example of using `lm-eval` to study the impact of prompts on model performance, taken from Muennighoff et al. (2022). Dots represent different prompts, as summarized by the histograms. Similar plots can be found in Workshop et al. (2023); Sanh et al. (2022); and others.

5.2 Difficulties in Comparing Scores Across Evaluation Setups

As mentioned in Section 2.3, scores on evaluation tasks can be substantially affected by the specifics of the evaluation implementation and setup. Here, we provide an example of studying such sensitivity to divergences in evaluation methodology, and how `lm-eval` can be used to improve confidence in the comparison of scores across models by preventing such divergences. We focus our attention on two popular language modeling benchmarks: the ARC question answering benchmark (Clark et al., 2018) and MMLU (Hendrycks et al., 2021).

	ARC Challenge		MMLU	
	Cloze	MMLU-style	MMLU-style	Hybrid
GPT-NeoX-20B	$38.0 \pm 2.78\%$	$26.6 \pm 2.53\%$	$24.5 \pm 0.71\%$	$27.6 \pm 0.74\%$
Llama-2-7B	$43.5 \pm 2.84\%$	$42.8 \pm 2.83\%$	$41.3 \pm 0.80\%$	$39.8 \pm 0.79\%$
Falcon-7B	$40.2 \pm 2.81\%$	$25.9 \pm 2.51\%$	$25.4 \pm 0.72\%$	$29.1 \pm 0.75\%$
Mistral-7B	$50.1 \pm 2.86\%$	$72.4 \pm 2.56\%$	$58.6 \pm 0.77\%$	$48.3 \pm 0.80\%$
Mixtral-8x7B	$56.7 \pm 2.84\%$	$81.3 \pm 2.23\%$	$67.1 \pm 0.72\%$	$59.7 \pm 0.77\%$

Table 1: Comparison of 0-shot model performance for several pretrained LMs (Black et al., 2022; Touvron et al., 2023; Penedo et al., 2023; Jiang et al., 2023; 2024) on ARC (Challenge subset) and MMLU across two commonly used prompt styles, evaluated using `lm-eval`. Unnormalized accuracy is reported, alongside 95% confidence intervals via SE of the mean.

ARC was first adapted to the in-context learning setting by Brown et al. (2020) who implement the dataset as a "cloze" task: the model is prompted with `Question: {question}\nAnswer:` and the likelihood of each potential completion string is compared. By contrast, MMLU (Hendrycks et al., 2020) provides the model with the question text, each of the 4 possible answers preceded by an answer letter A, B, C, or D, and scores the model based on the generation of the *letter corresponding to the correct answer*. Additionally, Hendrycks et al. (2020) aggregate scores via the micro average over all samples instead of the macro average over per-subject scores. However, not all papers evaluate on these tasks in the same way as the original formats.

However, if models do not adopt these approaches, or disclose their exact settings, it is impossible to reliably compare stated model performance. In Table 5.2, we compare evaluation on the Challenge subset of ARC using the prompt from Brown et al. (2020) ("Cloze") and using an MMLU-style answer letter with explicit multiple choice options ("MMLU-style")[8]. We additionally compare MMLU scores between the original MMLU prompting style ("MMLU-style") and an approach we term "Hybrid", consisting of an MMLU-style prompt but using the *answer strings* instead of answer letters as the set of continuations over which we can score. As described further in Appendix B.2, this can be done by modifying two lines in `lm-eval`'s ARC and MMLU config files.

Comparing these prompting styles, the degree to which models differ in performance widely varies, as well as *which* prompt performs better. If certain model creators chose one prompting and scoring style and certain other model creators chose the other, and each used the cited numbers from the respective technical reports for comparing their model to other baselines, the comparisons would be nonsensical and not provide information on which model were "truly" performant. Additionally, better statistical reporting such as the use of confidence intervals which we report, does not resolve these issues–while it gives a sense of how reliable a given measurement is (for example, MMLU has a smaller confidence interval due to the use of a larger amount of samples), it cannot tell us how much model performance will vary across different measurement settings and cannot indicate a comparison should not be made.

This demonstrates the vital importance of not copying numbers from across other papers' reported evaluation scores, and of sharing full details on one's own evaluation setup. We thus hope the use of `lm-eval` will boost rigor and confidence in novel evaluation results and encourage better communication of evaluation setups.

5.3 Empowering Benchmark Creators and LM Evaluation Research

Providing a library for evaluation *orchestration* that is configurable as we have described in Section 4 has many other uses, and we have observed the community leveraging `lm-eval` effectively for these purposes.

Experimentation on the complexity or difficulties in LM evaluation has been made easier via our configurable task design. Alzahrani et al. (2024) and Lyu et al. (2024) explore the effects of prompting and other distractors on model robustness and performance using `lm-eval`, as well as investigate the role of evaluation methodology such as the tradeoffs of loglikelihood versus generative evaluation, as we also detail in Appendix A.

`lm-eval` has been adopted by the community to make the design of novel benchmarks easier: our extensible Task configurations, and corresponding codebase have been used by the community to prototype the evaluation of their new benchmark datasets in `lm-eval`. By providing this location for community members to design and contribute novel evaluation code, we sidestep the challenging problem of tracking down and using extant evaluation code from various papers entirely: the *reference implementations* for these new tasks are directly in `lm-eval` in the first place. As described in Section 4.1, we strive to reduce barriers to task development and contribution, such as providing low-friction modes of development (modular configuration files) or examples implementing tasks "in the style of MMLU".

`lm-eval` has recently received contributions for a variety of datasets relying on `lm-eval` for their evaluation and benchmark prototyping and design (Faysse et al., 2024; Son et al., 2024a;b; Kweon et al., 2024; Li et al., 2024). By directly contributing their new evaluation tasks to `lm-eval`, benchmark authors also get to have full control over the *dissemination* (Hendrycks & Woodside, 2024) and implementation of their evaluation, making it far easier for the language modeling community to discover and recognize new benchmarking contributions that might otherwise go unrecognized or unadopted (Dehghani et al., 2021). This is the power of *orchestration*–the goal is to put new evaluation benchmarks in the hands

---

[8]While they do not report their evaluation set-up, this latter style appears to produce scores consistent with those reported in Jiang et al. (2024) for 25-shot ARC Challenge. We were unable to find a way to reproduce their scores using the standard cloze-style evaluation.

of the community, and put tools for creating benchmarks given an evaluation dataset in the hands of evaluation developers, while smoothing over potential roadblocks we discuss in Section 2. As an additional concrete example, tasks in `lm-eval` have been used to back not only the popular Open LLM Leaderboard (Beeching et al., 2023), but also the construction of arbitrary novel leaderboards, which have been used to make custom comparisons between models on more specific use cases, such as non-English languages.

Thus, the orchestration viewpoint we take allows downstream users and developers to create their own approaches which best fit their goals and applications, allowing for the fostering of more evaluation development and a more interoperable ecosystem, rather than setting a few chosen metrics in stone.

6 Conclusion

We have presented a number of common challenges in LM evaluation and our recommendations to mitigate the worst of these pitfalls. We introduce `lm-eval`, a library for evaluation orchestration built to enable easier and more reproducible benchmarking across common evaluation tasks and model implementations.

We hope that `lm-eval` will continue to be used by the community to improve rigor and our collective understanding of LM evaluations.

References

Gustaf Ahdritz, Nazim Bouatta, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, et al. Openfold: Retraining alphafold2 yields new insights into its learning mechanisms and capacity for generalization. *Biorxiv*, pp. 2022–11, 2022.

Mohammad Mahmudul Alam, Edward Raff, Stella Biderman, Tim Oates, and James Holt. Holographic global convolutional networks for long-range prediction tasks in malware detection. *arXiv preprint arXiv:2403.17978*, 2024.

Norah Alzahrani, Hisham Abdullah Alyahya, Yazeed Alnumay, Sultan Alrashed, Shaykhah Alsubaie, Yusef Almushaykeh, Faisal Mirza, Nouf Alotaibi, Nora Altwairesh, Areeb Alowisheq, M Saiful Bari, and Haidar Khan. When benchmarks are targets: Revealing the sensitivity of large language model leaderboards, 2024.

Jacopo Amidei, Paul Piwek, and Alistair Willis. Identifying annotator bias: A new IRT-based method for bias identification. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 4787–4797, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.421. URL https://aclanthology.org/2020.coling-main.421.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff, 2024.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language assistant as a laboratory for alignment, 2021.

Stephen H. 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 Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-Jian Jiang, and Alexander M. Rush. 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*, pp. 93–104, 2022.

Tessa Barton. Calibrating the mosaic evaluation gauntlet. `https://www.safe.ai/blog/devising-ml-metrics`, 2024.

Rachel Bawden and François Yvon. Investigating the translation performance of a large multilingual language model: the case of BLOOM. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, 2023. URL `https://arxiv.org/abs/2303.01911`.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. `https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard`, 2023.

Emily M. Bender and Batya Friedman. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604, 2018. doi: 10.1162/tacl_a_00041. URL `https://aclanthology.org/Q18-1041`.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

Stella Biderman. Magic: the gathering is as hard as arithmetic. *arXiv preprint arXiv:2003.05119*, 2020.

Stella Biderman and Edward Raff. Fooling moss detection with pretrained language models. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pp. 2933–2943, 2022.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. Pythia: A suite for analyzing large language models across training and scaling. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430. PMLR, 23–29 Jul 2023. URL `https://proceedings.mlr.press/v202/biderman23a.html`.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models*, 2022. URL `https://arxiv.org/abs/2204.06745`.

Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, Jennifer C Lai, and Robert L Mercer. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40, 1992.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. URL `https://arxiv.org/abs/2005.14165`.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of Bleu in machine translation research. In Diana McCarthy and Shuly Wintner (eds.), *11th*

Conference of the European Chapter of the Association for Computational Linguistics, pp. 249–256, Trento, Italy, April 2006. Association for Computational Linguistics. URL `https://aclanthology.org/E06-1032`.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or llms as the judge? a study on judgement biases. *ArXiv*, abs/2402.10669, 2024. URL `https://api.semanticscholar.org/CorpusID:267740522`.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023. URL `http://jmlr.org/papers/v24/22-1144.html`.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge, 2018.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pp. 177–190. Springer, 2005.

Ernest Davis. Benchmarks for automated commonsense reasoning: A survey. *ACM Computing Surveys*, 56(4):1–41, 2023.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models, 2024.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23.2, pp. 107–124, 2019.

Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022.

Manuel Faysse, Patrick Fernandes, Nuno M. Guerreiro, António Loison, Duarte M. Alves, Caio Corro, Nicolas Boizard, João Alves, Ricardo Rei, Pedro H. Martins, Antoni Bigata Casademunt, François Yvon, André F. T. Martins, Gautier Viaud, Céline Hudelot, and Pierre Colombo. Croissantllm: A truly bilingual french-english language model, 2024.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474, 2021. doi: 10.1162/tacl_a_00437. URL `https://aclanthology.org/2021.tacl-1.87`.

Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models, 2023.

Leo Gao. Multiple Choice Normalization in LM Evaluation. `https://blog.eleuther.ai/multiple-choice-normalization/`, 2021.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: an 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020. URL `https://arxiv.org/abs/2101.00027`.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021. URL `https://doi.org/10.5281/zenodo.5371628`.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pp. 1–9, 2007.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=uYLFoz1vlAC`.

R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, volume 7, pp. 785–794, 2006.

Dan Hendrycks and Thomas Woodside. Devising ml metrics. `https://www.safe.ai/blog/devising-ml-metrics`, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

Valentin Hofmann, Pratyusha Ria Kalluri, Dan Jurafsky, and Sharese King. Dialect prejudice predicts ai decisions about people's character, employability, and criminality. *arXiv preprint arXiv:2403.00742*, 2024.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration, 2020.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. Surface form competition: Why the highest probability answer isn't always right, 2022.

Tom Hosking, Phil Blunsom, and Max Bartolo. Human feedback is not gold standard, 2024.

Xinyu Hu, Mingqi Gao, Sen Hu, Yang Zhang, Yicheng Chen, Teng Xu, and Xiaojun Wan. Are LLM-based Evaluators Confusing NLG Quality Criteria? *ArXiv*, abs/2402.12055, 2024. URL https://api.semanticscholar.org/CorpusID:267750948.

F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63, 08 2005. ISSN 0001-4966. doi: 10.1121/1.2016299. URL https://doi.org/10.1121/1.2016299.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024.

Keller Jordan. Calibrated chaos: Variance between runs of neural network training is harmless and inevitable. In *The Twelfth International Conference on Learning Representations*, 2023.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 252–262, 2018.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models, 2024.

Sunjun Kweon, Byungjin Choi, Minkyu Kim, Rae Woong Park, and Edward Choi. Kormedmcqa: Multi-choice question answering benchmark for korean healthcare professional licensing examinations, 2024.

Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. In Heike Adel and Shuming Shi (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 175–184, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-demo.21. URL `https://aclanthology.org/2021.emnlp-demo.21`.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1192–1202, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1127. URL `https://aclanthology.org/D16-1127`.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, Gabriel Mukobi, Nathan Helm-Burger, Rassin Lababidi, Lennart Justen, Andrew B. Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhrugu Bharathi, Adam Khoja, Zhenqi Zhao, Ariel Herbert-Voss, Cort B. Breuer, Samuel Marks, Oam Patel, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Liu, Adam A. Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Russell Kaplan, Ian Steneker, David Campbell, Brad Jokubaitis, Alex Levinson, Jean Wang, William Qian, Kallol Krishna Karmakar, Steven Basart, Stephen Fitz, Mindy Levine, Ponnurangam Kumaraguru, Uday Tupakula, Vijay Varadharajan, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, and Dan Hendrycks. The wmdp benchmark: Measuring and reducing malicious use with unlearning, 2024.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models, 2023.

Valentin Li'evin, Christoffer Egeberg Hother, and Ole Winther. Can large language models reason about medical questions? *Patterns*, 5, 2022. URL `https://api.semanticscholar.org/CorpusID:250627547`.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL `https://aclanthology.org/2022.acl-long.229`.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation, 2023a.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, Singapore, December 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL `https://aclanthology.org/2023.emnlp-main.153`.

Yixin Liu, Alexander R Fabbri, Jiawen Chen, Yilun Zhao, Simeng Han, Shafiq Joty, Pengfei Liu, Dragomir Radev, Chien-Sheng Wu, and Arman Cohan. Benchmarking generation and evaluation capabilities of large language models for instruction controllable summarization. *arXiv preprint arXiv:2311.09184*, 2023c.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, et al. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. *arXiv preprint arXiv:2305.13169*, 2023.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.556. URL `https://aclanthology.org/2022.acl-long.556`.

Chenyang Lyu, Minghao Wu, and Alham Fikri Aji. Beyond probabilities: Unveiling the misalignment in evaluating large language models, 2024.

Ian Magnusson, Akshita Bhagia, Valentin Hofmann, Luca Soldaini, Ananya Harsh Jha, Oyvind Tafjord, Dustin Schwenk, Evan Pete Walsh, Yanai Elazar, Kyle Lo, Dirk Groeneveld, Iz Beltagy, Hannaneh Hajishirzi, Noah A. Smith, Kyle Richardson, and Jesse Dodge. Paloma: A benchmark for evaluating language model fit, 2023.

Benjamin Marie. Comparing the uncomparable to claim the state of the art: A concerning trend. Blog post, August 2022.

Benjamin Marie, Atsushi Fujita, and Raphael Rubino. Scientific credibility of machine translation research: A meta-evaluation of 769 papers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7297–7306, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.566. URL `https://aclanthology.org/2021.acl-long.566`.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Samuel Messick. Standards-based score interpretation: Establishing valid grounds for valid inferences. *ETS Research Report Series*, 1994(2):291–305, 1994. doi: https://doi.org/10.1002/j.2333-8504.1994.tb01630.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2333-8504.1994.tb01630.x`.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

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

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. State of what art? a call for multi-prompt LLM evaluation, 2024.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*, 2022.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL `https://aclanthology.org/P02-1040`.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, et al. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *arXiv preprint arXiv:2404.05892*, 2024.

Mohammad Taher Pilehvar and Jose Camacho-Collados. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*, 2018.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pp. 28043–28078. PMLR, 2023.

Ofir Press, Noah A. Smith, and Mike Lewis. Shortformer: Better language modeling using shorter inputs. *CoRR*, abs/2012.15832, 2020. URL `https://arxiv.org/abs/2012.15832`.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization, 2020.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. Hgrn2: Gated linear rnns with state expansion, 2024a.

Zhen Qin, Songlin Yang, and Yiran Zhong. Hierarchically gated recurrent neural network for sequence modeling. *Advances in Neural Information Processing Systems*, 36, 2024b.

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

Inioluwa Deborah Raji, Emily M. Bender, Amandalynne Paullada, Emily Denton, and Alex Hanna. AI and the Everything in the Whole Wide World Benchmark, 2021.

Inioluwa Deborah Raji, I. Elizabeth Kumar, Aaron Horowitz, and Andrew Selbst. The fallacy of ai functionality. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22. ACM, June 2022. doi: 10.1145/3531146.3533158. URL `http://dx.doi.org/10.1145/3531146.3533158`.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pp. 5389–5400. PMLR, 2019.

Joshua Robinson, Christopher Michael Rytting, and David Wingate. Leveraging large language models for multiple choice question answering, 2023.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*, 2011.

Tord Romstad, Marco Costalba, Joona Kiiski, Gary Linscott, Yu Nasu, Motohiro Isozaki, and Hisayori Noda. Stockfish, 2008. URL `https://github.com/official-stockfish/Stockfish`.

Sebastian Ruder. Challenges and Opportunities in NLP Benchmarking. `http://ruder.io/nlp-benchmarking`, 2021.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*, 2022.

Naomi Saphra, Eve Fleisig, Kyunghyun Cho, and Adam Lopez. First tragedy, then parse: History repeats itself in the new era of large language models. *arXiv preprint*, 2023.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage?, 2023.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting, 2023.

Thibault Sellam, Steve Yadlowsky, Ian Tenney, Jason Wei, Naomi Saphra, Alexander D'Amour, Tal Linzen, Jasmijn Bastings, Iulia Raluca Turc, Jacob Eisenstein, Dipanjan Das, and Ellie Pavlick. The multiBERTs: BERT reproductions for robustness analysis. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=K0E_F0gFDgA`.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. Large language models are not yet human-level evaluators for abstractive summarization. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 4215–4233, Singapore, December 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.278. URL `https://aclanthology.org/2023.findings-emnlp.278`.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. *arXiv preprint arXiv:2308.03825*, 2023b.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Guijin Son, Hanwool Lee, Sungdong Kim, Seungone Kim, Niklas Muennighoff, Taekyoon Choi, Cheonbok Park, Kang Min Yoo, and Stella Biderman. Kmmlu: Measuring massive multitask language understanding in korean, 2024a.

Guijin Son, Hanwool Lee, Suwan Kim, Huiseo Kim, Jaecheol Lee, Je Won Yeom, Jihyu Jung, Jung Woo Kim, and Songseong Kim. Hae-rae bench: Evaluation of korean knowledge in language models, 2024b.

Aarohi Srivastava, Abhinav Rastogi, Abhishek B Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Annasaheb Rahane, Anantharaman S. Iyer, Anders Johan Andreassen, Andrea Santilli, Andreas Stuhlmuller, Andrew M. Dai, Andrew D. La, Andrew Kyle Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakas, Bridget R. Roberts, Bao Sheng Loe, Barret Zoph, Bartlomiej Bojanowski, Batuhan Ozyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Stephen Howald, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, C'esar Ferri Ram'irez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Tatiana Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Daniel H Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Gonz'alez, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, D. Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth P. Donoway, Ellie Pavlick, Emanuele Rodolà, Emma FC Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fan Xia, Fatemeh Siar, Fernando Mart'inez-Plumed, Francesca Happ'e, François Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-L'opez, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Han Sol Kim, Hannah Rashkin, Hanna Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hubert Wong, Ian Aik-Soon Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, John Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, J. Brooker Simon, James Koppel, James Zheng, James Zou, Jan Koco'n, Jana Thompson, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jenni Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Oluwadara Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Jane W Waweru, John Burden, John Miller, John U. Balis, Jonathan Berant, Jorg Frohberg, Jos Rozen, José Hernández-Orallo, Joseph Boudeman, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Ochieng' Omondi, Kory Wallace Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Luca Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Col'on, Luke Metz, Lutfi Kerem cSenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Madotto Andrea, Maheen Saleem Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, M Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew Leavitt, Matthias Hagen, M'aty'as Schubert, Medina Baitemirova, Melissa Arnaud, Melvin Andrew McElrath, Michael A. Yee, Michael Cohen, Mi Gu, Michael I. Ivanitskiy, Michael Starritt, Michael Strube, Michal Swkedrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Monica Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, T MukundVarma, Nanyun Peng, Nathan Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas S. Roberts, Nicholas Doiron, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha Iyer, Noah Constant, Noah Fiedel,

Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter W. Chang, Peter Eckersley, Phu Mon Htut, Pi-Bei Hwang, P. Milkowski, Piyush S. Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, QING LYU, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ram'on Risco Delgado, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan Lebras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib J. Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Sam Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Sameh Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Kumar Reddy, Sneha Priscilla Makini, Soo hwan Lee, Spencer Bradley Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Rose Biderman, Stephanie C. Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq A. Ali, Tatsuo Hashimoto, Te-Lin Wu, Theo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, T. N. Kornev, Timothy Telleen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler O. Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay V. Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, W Vossen, Xiang Ren, Xiaoyu F Tong, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yang Song, Yasaman Bahri, Ye Ji Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yu Hou, Yuntao Bai, Zachary Seid, Zhao Xinran, Zhuoye Zhao, Zi Fu Wang, Zijie Jay Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. In *arXiv preprint 2206.04615*, 2022. URL `https://arxiv.org/abs/2206.04615`.

Arjun Subramonian, Xingdi Yuan, Hal Daumé III, and Su Lin Blodgett. It takes two to tango: Navigating conceptualizations of NLP tasks and measurements of performance. *arXiv preprint arXiv:2305.09022*, 2023.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

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

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=qVyeW-grC2k`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor,

Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019a.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *In the Proceedings of ICLR.*, 2019b.

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *ArXiv*, abs/2305.17926, 2023. URL https://api.semanticscholar.org/CorpusID:258960339.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. What language model architecture and pretraining objective work best for zero-shot generalization?, 2022.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Wenjin Yao, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. PandaLM: An automatic evaluation benchmark for LLM instruction tuning optimization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5Nn2BLV7SB.

Lucas Weber, Elia Bruni, and Dieuwke Hupkes. The ICL consistency test, 2023.

Albert Webson and Ellie Pavlick. Do prompt-based models really understand the meaning of their prompts? In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2300–2344, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.167. URL https://aclanthology.org/2022.naacl-main.167.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.

Johnny Tian-Zheng Wei and Robin Jia. The statistical advantage of automatic NLG metrics at the system level, 2021.

BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb,

Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Daniel McDuff, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model, 2023.

Minghao Wu and Alham Fikri Aji. Style over substance: Evaluation biases for large language models. *arXiv preprint arXiv:2307.03025*, 2023.

Shuyi Xie, Wenlin Yao, Yong Dai, Shaobo Wang, Donlin Zhou, Lifeng Jin, Xinhua Feng, Pengzhi Wei, Yujie Lin, Zhichao Hu, et al. Tencentllmeval: a hierarchical evaluation of real-world capabilities for human-aligned llms. *arXiv preprint arXiv:2311.05374*, 2023.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. A critical evaluation of evaluations for long-form question answering. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3225–3245, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.181. URL `https://aclanthology.org/2023.acl-long.181`.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training, 2024.

Zheng Xin Yong, Hailey Schoelkopf, Niklas Muennighoff, Alham Fikri Aji, David Ifeoluwa Adelani, KHALID ALMUBARAK, M Saiful Bari, Lintang Sutawika, Jungo Kasai, Ahmed Baruwa, et al. Bloom+1: Adding language support to bloom for zero-shot prompting. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. Evaluating large language models at evaluating instruction following. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=tr0KidwPLc`.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, Will Song, Tiffany Zhao, Pranav Raja, Dylan Slack, Qin Lyu, et al. A careful examination of large language model performance on grade school arithmetic. *arXiv preprint arXiv:2405.00332*, 2024.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *CoRR*, abs/2306.05685, 2023. doi: 10.48550/ARXIV.2306.05685. URL `https://doi.org/10.48550/arXiv.2306.05685`.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books, 2015.

A Formalizing Measurements

Here we provide a formal description of the most common approaches to obtaining outputs or measurements from LMs for evaluation, as we implement in `lm-eval`. We include this for a number of reasons: as a reference for future work, notes on the history of certain LM eval practices, and as an illustrative example of just how many implementation details or methodological choices *do not* typically make it into evaluation papers and yet can vitally impact results or findings.

A.1 Preliminaries

Throughout, we consider an auto-regressive language model (LM), with vocabulary V. Given an input consisting of tokens $x_0, x_1, ..., x_{n-1}$, the model outputs a probability distribution over the vocabulary, $P(x_n|x_0, x_1, ..., x_{n-1})$. Internally, this is represented as returning "logits" of shape $(1, |V|)$, which when taking a log-softmax over the vocabulary dimension, yields *log probabilities* ("logprobs" or "loglikelihoods") of each token in V. Logits are the raw, unnormalized predictions of the model before applying the softmax function. Crucially, due to the parallel training and causal masking of autoregressive LMs, it is possible to obtain from a single LM call with $x_0, x_1, ..., x_{n-1}$ as input, logits of shape $(n, |V|)$ with the i-th element of these logits representing $P(x_i|x_0, x_1, ..., x_{i-1})$ for all $1 \leq i \leq n$. (That is, for every token position of the input, we obtain concurrently the model's prediction for the subsequent token, starting from its prediction for the value of x_1 and ending with the model's predictions for the (not provided) "x_n" token.)

A.2 Ranking-Based Multiple Choice QA

Given our language model, we aim to compute the conditional (log) probability (or "log-likelihood") of a target string y conditioned on input x, denoted as $\log P(y|x)$. This can be performed in a single LM call.

Let $x = x_0, x_1, ..., x_{n-1}$ be an input sequence of n tokens and $y = y_0, y_1, ..., y_{m-1}$ be the target sequence of m tokens, where x_i and y_i represent individual tokens. To compute $\log P(y|x)$, we follow these steps:

1. Concatenate x and y to form a new sequence, but discard the final token y_{m-1}. The resulting sequence is $x_0, x_1, ..., x_{n-1}, y_0, y_1, ..., y_{m-2}$.

2. Pass this concatenated sequence through the language model to obtain logits l of shape $(n + m - 1, |V|)$, where $|V|$ is the size of the vocabulary. The last m positions in these logits correspond to the predicted probability distributions for the target tokens y_0 to y_{m-1}, conditioned on the input x and the preceding target tokens.

3. Apply a log-softmax function to the last m logits to obtain log probabilities for the completion tokens only.

4. Calculate the conditional loglikelihood of the target string y given the input x by summing the log probabilities of each target token:

$$\log P(y|x) = \sum_{i=0}^{m-1} \log p(y_i|x, y_0, ..., y_{i-1}) = \sum_{i=0}^{m-1} l(n + i, y_i), \qquad (1)$$

where $\log p(y_i|x, y_0, ..., y_{i-1})$ is the log probability of the i-th target token conditioned on the full input x and the preceding target tokens. (and where $x, y_0, ...y_{-1}$ denotes conditioning on only x.)

With this primitive for computing $\log P(y|x)$, several options for evaluation (and decisions regarding hyperparameters) become available.

Multiple Choice QA

Equation 1 determines how to compute $\log P(y|x)$. We now describe how to perform *loglikelihood-based multiple choice* as described by Brown et al. (2020): given k possible answer strings $a_1, a_2, ..., a_k$, we compute the model's answer to be $\texttt{argmax}(\log P(a_1|x), \log P(a_2|x), ..., \log P(a_k|x))$. In other words, the model selects the answer string with the highest conditional log probability given the input x.

This can be performed with worst-case k LM calls using the approach to calculate $\log P(y|x)$ for each $a_i = y$ described above. However, the number of LM calls can be reduced if one or more answer strings are only a single token in length. Assume some a_i is only encoded by a single token z. Then, when calculating the loglikelihood of another answer string a_0, we obtain the (log-softmaxed) logits of shape $(n + m - 1, |V|)$ as an intermediate output. These logits contain the predicted log probabilities for each token in the vocabulary at each

position, conditioned on the input x and the preceding tokens. To extract the loglikelihood of predicting the single-token answer a_i conditioned on x, we can simply select the element in l corresponding to token z at position n. This logit represents the log probability of predicting token z immediately after the input sequence $x_0, x_1, ..., x_{n-1}$.

Thus, we can calculate the loglikelihood of a single-token continuation "for free" and remove an additional LM call for each such single-token a_i.

Normalization

While the above approach uses the raw loglikelihoods of each given answer choice to select a model answer, other options are available. For instance, if each answer string a_i is different in length, this process may frequently default to selecting the shortest a_i simply because loglikelihoods are the sum over individual tokens' log probabilities. Several options for *normalizing* these loglikelihoods are possible, as also described in Gao (2021):

- **Token-length normalization**: each a_i's loglikelihood is divided by m_i, its length in tokens, to gain the per-token loglikelihood of each answer. This approach requires no additional LM calls, and is used alternately with raw loglikelihoods for most tasks by Brown et al. (2020).

- **Byte-length normalization**: each a_i's loglikelihood is divided by its length in bytes, removing the dependence on the model's tokenizer but still normalizing by answer string length. `lm-eval` provides this metric where applicable as `acc_norm`.

- **Mutual Information**: each a_i's loglikelihood is defined as $\log P(a_i|x) - \log P(a_i|null)$, where *null* is either the empty string, a BOS token, or a placeholder such as `"Answer:"`. This can be thought of as a notion of the *pointwise mutual information* (Shannon, 1948; Askell et al., 2021), $\log \left(\frac{P(a_i|x)}{P(a_i)} \right)$, which measures the increase in the likelihood of outputting a_i when conditioned on the input x, compared to the likelihood of outputting a_i unconditionally. Intuitively, this measure of mutual information captures the extent to which introducing x makes a_i more likely. Although this approach is nonstandard, it is provided in `lm-eval` under the option `acc_mutual_info`, and used selectively by Brown et al. (2020) and Askell et al. (2021) for certain tasks.

Computing Exact Match

In addition to computing loglikelihoods and normalized loglikelihoods, we may also want to determine whether a given target string y would be produced by greedily decoding from the input x. Let z be the concatenation of x and y as defined in the previous sections, and let l be the logits of shape $(n + m - 1, |V|)$ obtained by passing z through the language model. To compute the exact match, we compute $\sum_{i=0}^{m-1} \mathbb{1}[y_i = \texttt{argmax}(l(n + i, \cdot))]$, where $\mathbb{1}[\cdot]$ is the indicator function that returns 1 if the condition is true and 0 otherwise, and $l(n + i, \cdot)$ represents the logits vector corresponding to the model's output logits predicting the $n + 1 + i$-th token and therefore the i-th token position in y (0-indexed). Intuitively, this sum checks whether each token y_i in the target string y matches the most probable (argmax) token predicted by the model at each step of greedy decoding. If the sum equals m (the length of y), it means that all tokens in y would be produced by greedily generating m tokens starting from x. In this case, we return True to indicate an exact match. Otherwise, if the sum is less than m, we return False, indicating that y would not be produced by greedy decoding. Computing the exact match can be useful in scenarios where we want to assess whether the model can generate a specific target string verbatim.

Tokenization

In the above derivations, we assume that one can safely tokenize x and y separately and concatenate their tokenizations. This assumption is not always valid, and most widely-used language model tokenizers provide no such guarantees.

While this factor does not impact the validity of the above calculations, we note that this implies one should be very careful with how their prompt will be tokenized, especially in cases where the tokenization of an input and output pair separately may not align with the

tokenization the LM most often saw during training. There are some recently proposed mitigations to remedy this issue, such as "token healing"[9],

In the case of `lm-eval`, we achieve a majority of benefits via shifting trailing prompt whitespace into each target string y [10], and do not include trailing input whitespace for all tasks we implement, so this operation should be null for the vast majority of cases.

We hope that future work can examine the most practical way to remove such tokenization-based concerns and difficulties from evaluation, such as BPE dropout (Provilkov et al., 2020), other regularization techniques, or other novel tokenization innovations.

A.3 Perplexity evaluation

A common approach to measure language modeling performance on some data distribution D is to measure *perplexity*, which is defined as the exponential of the average negative loglikelihood per token (Jelinek et al., 2005; Brown et al., 1992), that is:

$$PPL = \exp\left(\frac{-1}{\sum_{j=1}^{|D|} N_j} \sum_{j=1}^{|D|} \sum_{i=1}^{N_j} \log P(y_{j_i}|y_{j_1}, \ldots, y_{j_{i-1}})\right), \tag{2}$$

where $|D|$ is the number of documents in the dataset, y_j is the j-th document in D, N_j is the total number of tokens in y_j, and y_{j_i} represents the i-th token of y_j.

To calculate perplexity on a selected dataset D, each dataset document y is tokenized and fed into a language model following the procedure to calculate $\log P(y)$ described in Appendix A.2, via computing $\log P(y|x)$, where x is set to either the empty string or a beginning-of-text token. Thus, given $\log P(y)$, for each document $y \in D$ we can sum up the per-document loglikelihoods and divide by the number of total dataset tokens. However, comparing perplexity across models that use different tokenizers can be challenging, as the number of tokens per document and the average next-token prediction difficulty will vary.

Tokenization

To avoid introducing a dependence on tokenizers while reporting perplexity scores, several options are available:

- **Bits per Byte:** This metric measures the average number of bits required to encode each byte of the input text, providing a tokenization-agnostic measure of language modeling performance (Gao et al., 2020). Formally:

$$BPB = \frac{-1}{log(2)} \left(\frac{-1}{\sum_{j=1}^{|D|} B_j} \sum_{j=1}^{|D|} \sum_{i=1}^{N_j} \log P(y_{j_i}|y_{j_1}, \ldots, y_{j_{i-1}})\right), \tag{3}$$

  where log is in base e and B_j is the length in bytes of document y_j. Alternately, bits per byte can be written as

$$BPB = \frac{\sum_{j=1}^{|D|} N_j}{\sum_{j=1}^{|D|} B_j} \log_2(PPL) = \frac{\sum_{j=1}^{|D|} N_j}{\sum_{j=1}^{|D|} B_j} \frac{\log(PPL)}{\log(2)}. \tag{4}$$

  That is, taking the base-2 log of perplexity and renormalizing by the number of bytes rather than tokens.

- **Word-Level Perplexity:** By tokenizing the input text into words, such as via splitting on whitespace, we can calculate perplexity based on the average loglikelihood per *word* rather than per-token, making the metric comparable across models with different subword tokenizers.

---

[9]`https://github.com/guidance-ai/guidance/blob/main/notebooks/tutorials/token_healing.ipynb`

[10]`https://github.com/meta-llama/llama/issues/217#issuecomment-1774147331`

- **Byte-level Perplexity:** Similarly, calculating perplexity averaged over the number of *bytes* instead allows for a different tokenization-independent perplexity calculation, as the number of bytes in each document's string remains constant regardless of the tokenizer used.

Both byte- and word-level perplexities can be calculated via replacing N_j in Equation A.3 instead with the number of bytes or "words" in document j.

In `lm-eval` we implement and report all 3 of the above metrics and report them as tokenization-agnostic measures of perplexity. This approach aligns with the work of Gao et al. (2020), who popularized the use of bits per byte for measuring perplexity, and has been adopted in subsequent studies such as Magnusson et al. (2023) and Hoffmann et al. (2022).

Sliding Window

Another challenge is the approach taken to measure perplexity on documents longer than the context length of a given LM. A natural approach, as used by Gao et al. (2020), is to chunk documents longer than a model's training context size L into non-overlapping chunks. For example, a document of length 4500 tokens evaluated using a model with context length 2048 would be processed as follows: tokens 0:2047 (with token 0 being a prepended BOS token) are fed to predict tokens 1:2048, then tokens 2048:4095 are fed to predict tokens 2049:4096, and finally tokens 4096:4499 are fed to predict tokens 4097:4500. The loglikelihoods of each chunk are then summed to obtain the entire document's loglikelihood.

However, Press et al. (2020) observe a phenomenon they call the "Early Token Curse", referring to the fact that tokens with a greater amount of context preceding them are fundamentally easier to predict, whereas the first several tokens a model must predict "from scratch" are difficult or impossible to predict without information to condition on. To mitigate this issue, they propose a *strided* or *sliding window* perplexity evaluation method.

Instead of creating non-overlapping windows of tokens of size L, the strided approach introduces a stride s such that overlapping windows of size L, shifting at each time by s positions, are used to score s new tokens' loglikelihoods. This is equivalent to Gao et al. (2020)'s approach when $s = L$. This approach reduces the skew of perplexity favoring models with larger L (and thus fewer tokens appearing at the beginning of a context window) that is introduced by the early token curse via reducing the number of tokens appearing with little context preceding them. However, it is worth noting that the prevalence of such affected tokens decreases with larger context window sizes L for a model.

A downside of the above method is that a naive implementation requires in the worst case, $\frac{L}{s}$ times the calls to an LM and $\frac{L}{s}$ the compute compared to the non-overlapping window approach. (However, some architectures can leverage KV cache reuse to avoid the cost of repeatedly re-encoding tokens). `lm-eval` follows (Gao et al., 2020) in using non-overlapping windows of size L. We believe this choice balances the computational cost and the mitigation of the early token curse, while still providing a standardized and comparable measure of language modeling performance.

A.4   Generative Evaluation

While loglikelihood-based tasks, such as multiple-choice question answering, provide a valuable measure of a language model's understanding and ability to rank given options, they do not directly assess the model's capacity to generate coherent and relevant text. Generative tasks, on the other hand, require the model to produce original text based on the given context.

Generative tasks have gained significant importance in recent times, particularly due to the fact that many popular language model APIs either do not provide[11] or greatly limit access[12] to log probabilities or other intrinsic measures of the model's confidence in its

---

[11]https://docs.anthropic.com/claude/reference/complete_post
[12]https://platform.openai.com/docs/api-reference/chat

outputs. This shift has made it especially necessary to rely on the generated text itself to evaluate the model's performance and capabilities.

Sampling Hyperparameters

Generative tasks often involve various techniques for controlling the diversity and quality of the generated text, such as sampling with temperature, top-k or top-p (nucleus) sampling (Holtzman et al., 2020), and beam search (Li et al., 2016). The choice of these hyperparameters can significantly impact the model's output and, consequently, its performance on the task. It is essential to consider and report these hyperparameters when evaluating generative models, as they can greatly influence the generated text's characteristics and the model's overall performance.

Scoring and Answer Extraction

Due to the challenges in measuring language model output (Section 2.1), particularly in verifying the semantics of natural language, and because free-form generation sacrifices the benefit of the artificially restricted input space of multiple-choice tasks, the challenge of scoring answers for quality or correctness must be tackled differently.

The open-ended nature of generative tasks means that there may be multiple valid and appropriate responses to a given prompt. A common evaluation strategy is to use few-shot prompts, where the model is provided with a number of examples demonstrating the desired input-output format. The model is then prompted with a new input, and its generated response is extracted using regular expressions (regex) or other heuristic approaches to obtain the normalized answer strings which can be evaluated using exact-match or other metrics. This approach allows for a more structured evaluation of the model's ability to generate accurate and relevant responses based on the given examples.

However, this is often a highly imperfect solution, as different models may generate responses in varying formats, making it challenging to create a universal regex pattern that works for all models. Moreover, the effectiveness of the regex-based extraction is highly dependent on the specific format used in the original task creation, which could introduce bias towards models that generate responses in a similar format. To address these limitations, `lm-eval` provides a highly customized answer extraction mechanism through a **Filter** component tied to **Task** implementations, allowing the model output to be put through an arbitrary number of filters and post-processing steps.

These custom heuristic approaches make the release of evaluation code, and our recommendations in general, even more crucial. Without knowledge of the extent to what extraction code is used, how it may be tailored to a model, or without access to model outputs, it is difficult to separate models' compliance with the evaluation *format* from their answer *correctness*. Therefore, it is essential to provide detailed information about the answer extraction process and make the code and model outputs available to ensure transparency and reproducibility in generative model evaluation.

A.5   Comparing Generative and Loglikelihood-based Evaluation

A notable advantage of generative evaluation is that it might serve as a better proxy for assessing a language model's performance in real-world applications. In most practical use cases, such as the increasingly popular conversational chatbot format, language models are expected to generate coherent and contextually relevant text based on a given prompt or context. By focusing on the quality and appropriateness of the generated text, generative evaluation provides a more direct assessment of the model's performance in these real-world use cases. This is in contrast to loglikelihood-based tasks, which, while informative, may not fully capture the model's ability to generate text that is both fluent and contextually appropriate.

On the other hand, loglikelihood-based evaluations have their own advantages, particularly when it comes to evaluating smaller or weaker models, or "base" models not trained to follow instructions (Sanh et al., 2022). These evaluations can provide a useful ranking or measurement of a model's performance, even if the model is not capable of generating high-

quality text on its own. By assessing how likely the model is to assign a high probability to the correct answer, loglikelihood-based evaluations can offer insights into the model's understanding of the task. Moreover, techniques like Brier Score can be used to obtain smoother measurements of a model's performance (Schaeffer et al., 2023), providing a more nuanced assessment of its capabilities. This can be particularly valuable when comparing and ranking models of different sizes and capacities.

B Case Studies

B.1 Case Study: Benchmarking Novel LM Architectures

Recent work has explored the potential of various novel architectural designs to enable fully-subquadratic complexity in input sequence length while still achieving transformer-level quality or better (See Table 2 for a number of references). However, tracking progress towards this goal requires a reliable set of evaluations that can 1) be used to compare fairly against baselines and 2) provide useful signal even at small scales of experimentation.

As shown in Section 5.2 and elsewhere in the literature, evaluating models on different prompts or differently-framed evaluation setups for the same evaluation "task" can render comparisons not meaningful. This is especially important in the case of small language models trained on novel architectures, as "weaker" models may be hypothetically less robust to evaluation noise or differences in evaluation setup.

`lm-eval` has been used as a tool by many recent architecture releases to evaluate the performance of their proposed architecture against common baselines. We survey a number of recent releases, and note, for a number of commonly used benchmarks, whether researchers report their architecture's performance on that benchmark, and if they specifically state the usage of `lm-eval` to evaluate these tasks where applicable.

The selection of tasks we check are the following:

Wikitext-103 ("Wiki") (Merity et al., 2016): Wikitext-103 is a 103 million word language modeling dataset sourced from Wikipedia by Merity et al. (2016) to serve as a language modeling benchmark. It contains a training, validation, and test split, with a typical setup being to train a (small) model from scratch on the dataset and evaluate its test set perplexity (PPL).

Long Range Arena ("LRA") (Tay et al., 2021): LRA is a sequence modeling dataset consisting of a suite of various tasks meant to test the long-range modeling abilities of models. While solving the more challenging longer-context tasks in LRA drove earlier work on long-context sequence models (Gu et al., 2022), subsequent work has shown that LRA may not correlate with desired downstream tasks for pretrained models (Alam et al., 2024).

PIQA ("PQ") (Bisk et al., 2020): PIQA is a question-answering dataset meant to evaluate physical common-sense reasoning. It is typically evaluated using loglikelihood-based multiple choice and normalized (`acc_norm`) or unnormalized (`acc`) accuracy is reported.

OpenBookQA ("OB") (Mihaylov et al., 2018): OpenBookQA is a question-answering dataset meant to evaluate the combination of common knowledge with open-book exam questions. It is also typically evaluated using loglikelihood-based multiple choice and normalized (`acc_norm`) or unnormalized (`acc`) accuracy is reported.

WinoGrande ("WG") (Sakaguchi et al., 2019): WinoGrande is a dataset consisting of Winograd Schema Challenge-like minimal sentence pairs with one word flipped. Language models are typically evaluated on this dataset by comparing the probability of correctly completing the end of the sentence given the correct or incorrect context (the sentence up to and including the flipped word), and reporting accuracy (`acc`) (Radford et al., 2019; Brown et al., 2020).

HellaSwag ("HS")   (Zellers et al., 2019): HellaSwag is an adversarially created dataset meant to test "commonsense natural language inference" mined from WikiHow. Models are typically evaluated by choosing the most likely to be generated completion text from a correct option and (nonsensical) set of incorrect answer options (`acc, acc_norm`).

AI2 Reasoning Challenge ("ARC")   (Clark et al., 2018): ARC is a challenging question answering dataset consisting of an Easy and Challenge subset. Questions are sourced from standardized tests on natural sciences. `lm-eval` follows Brown et al. (2020) in using a "cloze" style loglikelihood-based evaluation and reports `acc, acc_norm` over the set of answer strings.

LAMBADA ("LMB")   (Paperno et al., 2016): The LAMBADA dataset is a word prediction benchmark consisting of short passages from Book Corpus (Zhu et al., 2015) books, with a language model required to predict the final word. Radford et al. (2019) introduce a cleaned and detokenized variant of LAMBADA [13], often denoted as "Lambada (OpenAI)."

This corresponds to the `lambada_openai` task in `lm-eval`, and the dataset can be found at https://huggingface.co/datasets/EleutherAI/lambada_openai. Two metrics are reported: average perplexity over the continuation string, and exact match accuracy calculated directly as described in Appendix A.

SuperGLUE ("SGLUE")   (Wang et al., 2019a): SuperGLUE is a benchmark containing a collection of NLU tasks (BoolQ (Clark et al., 2019), CB (De Marneffe et al., 2019), COPA (Roemmele et al., 2011), MultiRC (Khashabi et al., 2018), ReCoRD (Zhang et al., 2018), RTE (Wang et al., 2019b; Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), WiC (Pilehvar & Camacho-Collados, 2018), WSC (Levesque et al., 2012)). `lm-eval` implements SuperGLUE as a loglikelihood-based multiple choice classification task based on (Brown et al., 2020).

Arch.	Wiki	LRA	PQ	OB	WG	HS	ARC	LMB	SGLUE
H3 (Fu et al., 2023)	○								◖
Hyena (Poli et al., 2023)	○							◖	◖
Pythia (Biderman et al., 2023)			●		●	●	●	●	
RWKV-4 (Peng et al., 2023)		○	●	●	●	●	●	●	
RetNet (Sun et al., 2023)			◖		◖	◖			
HGRN1 (Qin et al., 2024b)	○	○	●		●	●	●		●
Mamba (Gu & Dao, 2023)			●	●	●	●	●	●	
GLA (Yang et al., 2024)			●	●	●	●	●	●	
Based (Arora et al., 2024)			●	●	●	●	●	●	●
Griffin,Hawk (De et al., 2024)			◖		◖	◖	◖		
RWKV-5,6 (Peng et al., 2024)			●		●	●	◖	●	
HGRN2 (Qin et al., 2024a)	○	○	◖		◖	◖	◖		

Table 2: A number of released subquadratic language modeling architectures, alongside which of a selection of common evaluation tasks were chosen by the authors for evaluation. ● denotes that the `lm-eval` implementation of a task was reported to be used, while ◖ means that a task is supported by `lm-eval`, but it could not be determined from the paper or linked code whether `lm-eval` was used. ○ denotes a model was evaluated on this task, but it is not supported by `lm-eval`.

We report our assessments in Table 2. Overall, we find that `lm-eval` has been used frequently to measure new architectures' zero-shot performance, and increases our confidence that most new works are evaluating on the same key benchmarks and methodologies. We note that not using `lm-eval` does not imply a lack of evaluation rigor, and that encouragingly, many works which do not use `lm-eval` do however report implementation details, such as holding

---

[13]For more information, see here and here.

the tokenizer used for perplexity calculations on Wikitext-103 constant (Fu et al., 2023; Poli et al., 2023) and report hyperparameters. Additionally, public and reproducible evaluation code is not sufficient for full reproducibility and confident comparison–reporting dataset contents, or training one's own controlled baselines, is also important and frequently but not always done. However, we simply wish to emphasize the ease with which `lm-eval` provides tools to researchers for performing research on advancing language modeling.

B.2 Comparisons Across Evaluation Settings

Here we provide extra materials and information for Section 5.2.

Here, to illustrate an example of the configurability and reproducibility of our library, we share the configurations used to compare the effect of prompting on MMLU and ARC.

```yaml
group:
  - ai2_arc
task: arc_easy
dataset_path: allenai/ai2_arc
dataset_name: ARC-Easy
output_type: multiple_choice
training_split: train
validation_split: validation
test_split: test
doc_to_text: "Question: {{question}}\nAnswer:"
doc_to_target: "{{choices.label.index(answerKey)}}"
doc_to_choice: "{{choices.text}}"
metric_list:
  - metric: acc
    aggregation: mean
    higher_is_better: true
metadata:
  version: 1.0
```

A YAML configuration file for the ARC-easy task, implemented in the "cloze" style as done by (Brown et al., 2020).

```yaml
group:
  - ai2_arc
task: arc_easy_mmlu
dataset_path: allenai/ai2_arc
dataset_name: ARC-Easy
output_type: multiple_choice
training_split: train
validation_split: validation
test_split: test
doc_to_text: "{{question.strip()}}\n{% for choice in choices.text %}{{
    choices.label[loop.index - 1]}}. {{choice}}\n{% endfor %}Answer:"
doc_to_target: "{{choices.label.index(answerKey)}}"
doc_to_choice: "{{choices.label}}"
metric_list:
  - metric: acc
    aggregation: mean
    higher_is_better: true
metadata:
  version: 1.0
```

A YAML configuration file for the ARC-easy task, as implemented following the prompting style of MMLU in Hendrycks et al. (2020).

We can observe that these configuration files define several components:

- The source dataset from the `Datasets`(Lhoest et al., 2021) library (local datasets are also supported), and the splits to use for testing and few-shot examples. Few-shot examples are drawn from a special fewshot split if specified, else drawn from

the training set, validation set, or (in the worst case) non-overlapping test set examples with the current test set example being evaluated, in decreasing order of prioritization.

- the `doc_to_*` attributes define mappings to input prompt, gold target label, and the list of answer choice strings, respectively in order.

- We provide a list of metrics to use–here, `acc` denotes unnormalized loglikelihood to score answers, and `acc_norm` using byte-length normalization of loglikelihoods.

- Finally, the `metadata.version` field stores the task's version attribute to report.

For prototyping, and for the quick modification of interrelated task variants during experimentation, configurations can also inherit from one another: the following is a config file for ARC-challenge, in the cloze style:

```
include: arc_easy.yaml
task: arc_challenge
dataset_name: ARC-Challenge
```

a config file for a single MMLU subset in its original style is the following:

```
dataset_path: cais/mmlu
test_split: test
fewshot_split: dev
fewshot_config:
  sampler: first_n
output_type: multiple_choice
doc_to_text: "{{question.strip()}}\nA. {{choices[0]}}\nB. {{choices[1]}}\
    nC. {{choices[2]}}\nD. {{choices[3]}}\nAnswer:"
doc_to_choice: ["A", "B", "C", "D"]
doc_to_target: answer
metric_list:
  - metric: acc
    aggregation: mean
    higher_is_better: true
metadata:
  version: 0.0
```

And the "Hybrid" variant:

```
dataset_path: cais/mmlu
test_split: test
fewshot_split: dev
fewshot_config:
  sampler: first_n
output_type: multiple_choice
doc_to_text: "{{question.strip()}}\nA. {{choices[0]}}\nB. {{choices[1]}}\
    nC. {{choices[2]}}\nD. {{choices[3]}}\nAnswer:"
doc_to_choice: choices
doc_to_target: answer
metric_list:
  - metric: acc
    aggregation: mean
    higher_is_better: true
metadata:
  version: 0.0
```