PROXYQA: An Alternative Framework for Evaluating Long-Form Text Generation with Large Language Models

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have succeeded remarkably in **understanding** long-form contents. However, exploring their capability for **generating** long-form contents, such as reports and articles, has been relatively unexplored and inadequately assessed by existing benchmarks. The prevalent evaluation methods, which predominantly rely on crowdsourcing, are recognized for their labor-intensive nature and lack of efficiency, whereas automated metrics, such as the ROUGE score, demonstrate discordance with human judgment criteria. In this paper, we propose PROXYQA, an innovative framework dedicated to assessing long-text generation. PROXYQA comprises in-depth human-curated *meta-questions* spanning various domains, each accompanied by specific *proxy-questions* with pre-annotated answers. LLMs are tasked to generate extensive content in response to these meta-questions, by engaging an evaluator and incorporating the generated texts as contextual background, PROXYQA assesses the generated content's quality through the evaluator's accuracy in addressing the *proxy-questions*. We examine multiple LLMs, emphasizing PROXYQA's demanding nature as a high-quality assessment tool. Human evaluation demonstrates that the *proxy-question* method is notably self-consistent and aligns closely with human evaluative standards. The dataset and leaderboard will be publicly available.

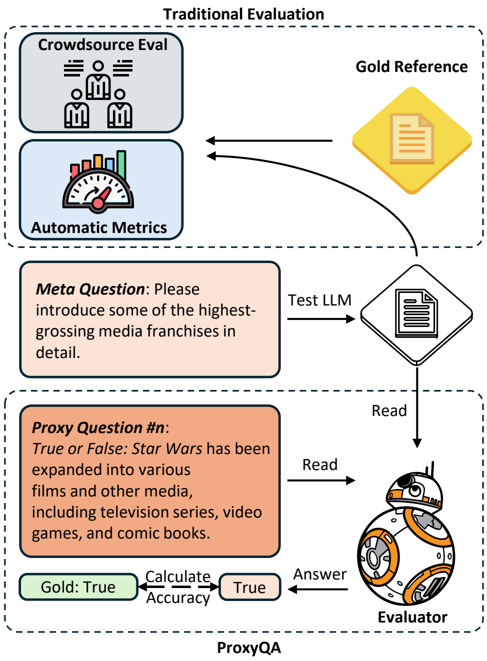

Figure 1: Prior efforts assess generated content by matching it with references through human evaluation or automated metrics. PROXYQA evaluates the knowledge coverage and informativeness by checking if generated contents contain sufficient information to answer a set of proxy questions.

1 Introduction

Recent Large Language Models (LLMs) have made significant advancements (Brown et al., 2020; Touvron et al., 2023a,b; OpenAI, 2022a, 2023b). GPU technology innovations and memory-efficient attention mechanisms (Dao et al., 2022; Dao, 2023) have further enabled LLMs to model context sequences spanning tens of thousands of tokens (Anthropic, 2023; OpenAI, 2023c), paving the way for sophisticated applications such as analyzing complex scientific essays and generating detailed reports. As long-context LLMs evolve, several benchmarks have emerged to evaluate their ability to handle extensive contexts (Shaham et al., 2023; Bai et al., 2023; An et al., 2023; Zhang et al., 2023). However, these assessments primarily focus on LLMs' **comprehension** of lengthy passages, using automated metrics to measure performance. This leaves a significant gap in understanding LLMs' proficiency in **generating** long-form texts, an essential aspect that requires further investigation.

One primary roadblock to understanding LLMs' capability to generate long-form texts is the lack of competent evaluation methods. Current methods, often involving a combination of automated metrics and crowdsourced annotations, leave much to be desired (Xu et al., 2023). For instance, automated metrics that use word-level string (Lin, 2004) or meaning representation matching (Yuan et al., 2021) rely on gold references, which unfortunately do not exist for many generation tasks, such as reports or essay writing. Furthermore, these automated metrics are inadequate for reliably assessing long-form content due to the considerable and unstructured space of potential outputs (Celikyilmaz et al., 2020; Krishna et al., 2021). Human evaluation has its own set of issues too. Crowdsourced workers may lack the necessary expertise for evaluating knowledge-rich content, and domain experts' subjective preferences could result in inconsistent evaluations (Xu et al., 2023). While recent studies explored using LLMs for evaluation (Chiang and Lee, 2023; Liu et al., 2023), LLMs have been found to lack the most current information required for precise verification. Moreover, their assessments have been observed to be inconsistent (Shen et al., 2023). There is a clear need for more robust and precise evaluation methods.

To address this issue, we introduce PROXYQA, a benchmark comprising human-curated meta-questions covering a wide range of subjects, from computer science to history. These meta-questions require domain expertise and up-to-date knowledge, prompting LLMs to generate detailed and comprehensive responses. To assess knowledge coverage and informativeness, we pair each meta-question with a series of proxy-questions and answers that capture its essential points. As illustrated in Figure 1, PROXYQA uses an evaluator to answer proxy-questions based on the long-form content produced by LLMs, rather than comparing the output to a reference. If the generated content is sufficiently detailed and accurate, it should equip the evaluator with enough information to thoroughly answer all associated proxy-questions.

PROXYQA offers several benefits. By employing proxy questions and an evaluator, it eliminates the need for direct comparison against a single gold reference, enabling a more accessible and subjective evaluation. Using this approach allows evaluators without specific domain knowledge to assess content. Additionally, unlike previous datasets compiled from online sources (Nguyen et al., 2016; Fan et al., 2019a) that potentially leading to data contamination (Sainz et al., 2023), all the *proxy-questions* and answers are invisible to public, thereby preventing data leakage. We apply PROXYQA to extensively test different LLMs (Touvron et al., 2023a,b; Taori et al., 2023; Chiang and Lee, 2023; OpenAI, 2022a, 2023b), including the LLMs that enhanced iterative reasoning (Yao et al., 2023a) and retrieval augmentation (Bing, 2023; Gemini, 2023). A systematic human evaluation demonstrates that PROXYQA offers a highly consistent evaluation scope, surpassing inter-human agreement rates while maintaining strong correlations with the majority preferences of humans.

2 Related Work

2.1 Long-Form Text Generation

Significant strides have been made in long-form text generation, particularly in story generation (Fan et al., 2019b; Xu et al., 2020), paragraph completion (Kang and Hovy, 2020), long-term conversation (Xu et al., 2022) and article generation (Hua and Wang, 2020; Hu et al., 2022). A closely related field is long-form question answering (Fan et al., 2019a; Dasigi et al., 2021; Stelmakh et al., 2022; Lee et al., 2023), which involves generating detailed responses to complex information-seeking questions. ELI5 (Fan et al., 2019a) was a pioneer dataset for generating explanatory paragraphs in response to open-ended questions, utilizing answers from Reddit. QASPER (Dasigi et al., 2021) and QASA (Lee et al., 2023) extend general factoid questions to the domain of scientific literature. Evaluating answers on these datasets relies on comparing the generated texts with the provided single reference. However, open-ended questions can be answered in myriad different ways. ASQA (Stelmakh et al., 2022) also introduces a set of disambiguated questions from AmbigQA (Min et al., 2023) for evaluating ambiguous questions. They assume that long-form answers to ambiguous questions should resolve ambiguity. In contrast, ProxyQA aims to gauge the informativeness and comprehensiveness of long-form answers, without being confined solely to ambiguous questions.

2.2 Text Generation Evaluation

Automated metrics such as surface form matching (Lin, 2004; Banerjee and Lavie, 2005) and semantic representation comparison (Zhang et al., 2020; Yuan et al., 2021), face challenges with long-form content due to their inability to handle the diversity of the potential outputs (Celikyilmaz et al., 2020; Krishna et al., 2021). They often do not align with human judgment (Xu et al., 2023). Attempts

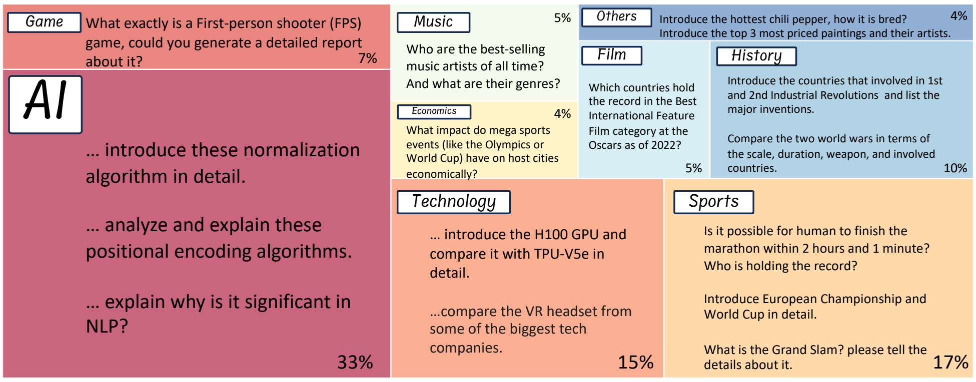

Figure 2: Meta-questions in PROXYQA cover various domains, such as AI research, historical event investigations, sports and entertainment analysis, and more.

to use LLMs for evaluation (Chiang and Lee, 2023; Liu et al., 2023) are hindered by their limited access to current information and inconsistency in performance (Shen et al., 2023). Evaluators also have difficulties, particularly if they lack expertise, which can impair their judgment on key dimensions like informativeness and factuality (Gillick and Liu, 2010; Iskender et al., 2020). Strategies to enhance human evaluations include A/B testing, as seen with HURDLES (Krishna et al., 2021) and WebGPT (Nakano et al., 2021), with the latter demonstrating that providing evidence helps annotators make more informed decisions. The list of proxy questions in PROXYQA can be viewed as evidence to assist the evaluator in making decisions. While some research focused on coherence (Goyal et al., 2022; Jiang et al., 2022; Deng et al., 2022) and factuality (Goyal and Durrett, 2020; Laban et al., 2022; Min et al., 2023) in related tasks like summarization, our work emphasizes informativeness and coverage.

3 A Long-form Generation Benchmark

An alternative framework for evaluating long-form text generation is created. 100 meta-questions are annotated to prompt LLMs to generate detailed and informative responses, which cover subjects in artificial intelligence (AI) research, historical event investigations, sports and entertainment event analysis. The topic distribution is shown in Figure 2. Each meta-question accompanies various *proxy-questions* with annotated answers, which are invisible to the LLMs to be tested. To proxy the objective evaluation of the generated contents to subjective metrics, we adopt an *evaluator* that takes the generated contents as contextual background and answers the proxy-questions. We assume that only if the contextual background is informative and comprehensive, can the proxy-questions be well-addressed by the evaluator. Therefore, the quality of the generated contents is reflected in the evaluator's accuracy on the proxy-questions.

3.1 Creating the Meta-questions

Meta-questions were manually raised by five experienced researchers, who were instructed to initiate meta-questions in areas with which they were most familiar or had a keen interest. Besides the most-concerned topics such as AI research, sports and gaming, less popular domains such as infrastructure and agriculture are also included. We focus on questions that are aligned with real-life scenarios and should be well-addressable in reports or articles. For instance, a pertinent question within the *Computer Science* domain could be: "Could you elaborate on the development of Model Parallelism and Pipeline Parallelism, detailing key milestones and contributions?" This meta-question aligns with interests in parallel computing techniques. In contrast, questions of the sort "Did Aristotle use a laptop?" from StrategyQA (Geva et al., 2021) are omitted due to their lack of occurrence in realistic settings.

Difficulty of Meta-question Meta-questions can be classified into two levels of difficulty: easy and hard. Easy questions can be sufficiently answered using only information from Wikipedia, while hard questions demand the integration of Wikipedia content with insights derived from a wider range of open-domain knowledge sources to formulate a comprehensive response. Generally, most LLMs can effectively address easy questions given their extensive training on Wikipedia corpora. Conversely, hard questions pose a challenge to the models' ability to acquire information beyond the commonly used pre-training data, necessitating access to specialized private corpora, web searches, or document retrieval for a comprehensive response. For more details, please refer to Appendix A.8.

3.2 Annotation of Proxy-Question

The evaluation of the generated reports is proxied to the evaluator's accuracy on proxy-questions. Experts are tasked with identifying the pivotal content that a satisfactory answer to a meta-question must contain. Then they craft a series of proxy-questions that probe these identified key points. For instance, regarding the example of model parallelism and pipeline parallelism mentioned in Section 3.1, a thorough answer should incorporate in-depth information about Gpipe (Huang et al., 2019), Megatron (Shoeybi et al., 2019), and other pertinent subjects. Therefore, annotators develop proxy-questions that specifically focus on Gpipe, Megatron and other related topics. As our destination is not to stress-test the evaluator but to quantify the quality of the generated contents, we present straightforward and concise proxy-questions, deliberately avoiding multi-hop and complex reasoning queries. Each annotated response is provided in a boolean format, ensuring that evaluators can effortlessly answer these proxy-questions, given a sufficiently high-quality generated context. Conversely, if an evaluator struggles to address these simple proxy-questions based on the provided context, it indicates a significant deficiency in the generated context, with crucial information being absent.

3.3 Quality Assurance

The meta-question and its corresponding proxy-questions were annotated and quality-checked iteratively. During the annotation, we excluded three meta-questions that were offensive, politically sensitive, ethically concerning, or not safe for work (NSFW). Meta-questions that have been previously posted and well-addressed on relevant forums are replaced. This ensures that LLMs cannot generate answers by directly copying content from these platforms, therefore preserving the integrity of our dataset. As the foundation of the evaluation in PROXYQA, proxy-questions are curated through a multi-round annotation process, where experts iteratively exchange the meta-questions while supplementing and verifying proxy-questions annotated by the others. Each meta-question is thus repeatedly given to different experts to label different proxy-questions, until a consensus is reached that all experts agree that the points covered by the proxy-questions are sufficiently comprehensive. Such an alternate labeling process ensures a multi-perspective rubric, leading to an experts-consolidated benchmark. After the iterative annotation, each meta-question is coupled with 15.5 proxy-questions on average, we then measure the inter-annotator agreement of PROXYQA. We randomly extracted a subset of 50 proxy-questions and tasked the annotators to re-annotate them. Following the Kazemi et al. (2021), the Randolph's free-marginal multi-rater κ (Randolph, 2010), an alternative to Fleiss' κ are measured. PROXYQA achieves $\kappa = 0.936$ thanks for the introduction of the iterative annotation process. Additionally, it is worth noting that real-world knowledge evolves over time. Therefore, to ensure the quality of PRXOYQA, our experts are required to review and update proxy-questions periodically, as detailed in the Appendix A.9.

3.4 Evaluators

To ensure the generated contents such as markdown tables and math formulas can be well encoded, GPT-4 and its variant, GPT-4-Turbo (GPT-4-1106-Preview), are utilized as evaluators. Instead of applying retrieval augmented generation, evaluators are required to read the generated contents and answer the proxy-questions. We prompt the evaluator to formulate answers to the proxy-questions strictly from the information presented within the contextual background. Evaluator's accuracy can be high only if the generated contextual background is informative and comprehensive enough. The objective assessment of these reports is subsequently anchored to the precision of the GPT evaluator. Moreover, we ensemble the evaluation results from GPT-4 and GPT-4-Turbo to reinforce the reliability and robustness of the assessment.

4 Experiments

4.1 Setup

Assessment and Models To measure LLMs' performance on PROXYQA, we compute the accuracy of the proxy-questions across easy and hard splits. Open-sourced LLMs such as LLaMA and its instruction-finetuned variants are tested. We also evaluated closed-sourced LLMs (e.g. GPT) and web-augmented LLMs, including Bard (Gemini Pro) (Gemini, 2023) and New Bing (Bing, 2023). Details of all the tested models are in the Ap-

	GPT-4			GPT-4-Turbo			Ensemble		
	Easy	*Hard*	*avg.*	*Easy*	*Hard*	*avg.*	*Easy*	*Hard*	*avg.*
Base LLaMA									
LLaMA-7B	5.25	0.68	3.05	5.89	1.23	3.64	5.57	0.96	3.34
LLaMA2-7B	4.74	0.55	2.72	5.38	0.55	3.04	5.06	0.55	2.88
LLaMA2-13B	**6.15**	**1.65**	**3.97**	**6.91**	**1.37**	**4.24**	**6.53**	**1.51**	**4.11**
Instruction-Finetuned LLaMA									
Alpaca-7B	12.42	5.62	9.14	14.60	9.33	12.05	13.51	7.48	10.60
Vicuna-13B	19.85	**17.15**	18.54	**22.66**	**21.26**	**21.99**	21.25	**19.20**	**20.26**
LLaMA2-7B-Chat	**21.25**	16.74	**19.07**	20.23	18.11	19.20	20.74	17.42	19.14
LLaMA2-13B-Chat	21.13	16.87	**19.07**	22.02	17.42	19.80	**21.57**	17.15	19.44
GPT APIs									
GPT-3.5-Turbo	25.61	21.40	23.57	26.12	22.36	24.30	25.87	22.97	23.94
GPT-4	30.35	23.05	26.82	30.35	23.55	27.55	30.35	23.80	27.19
GPT-4-Turbo	**35.21**	**31.69**	**33.50**	**34.83**	**33.88**	**34.37**	**35.02**	**32.78**	**33.94**
Web-Augmented LLMs									
ReAct (GPT-4)	20.74	13.72	17.35	20.49	13.17	16.95	20.61	13.44	17.15
ReAct (GPT-4-Turbo)	23.94	18.11	21.13	23.56	18.79	21.26	23.75	18.45	21.19
Bard (Gemini Pro)	26.63	22.22	24.50	25.48	25.51	25.50	25.06	23.87	25.00
New Bing (Creative Mode)	**39.56**	**37.72**	**38.67**	**40.33**	**39.78**	**40.06**	**39.95**	**38.75**	**39.37**

Table 1: Evaluation of various LLMs' knowledge coverage and informativeness on PROXYQA, accuracy on the easy and hard splits are reported.

pendix A.3. Each LLM is prompted with "Write a well-structured and extensive report to answer the question: [META-QUESTION]" under the setting of zero-shot evaluation. The max decoding length of open-sourced and closed-sourced LLMs is set to their reported maximum length. All the other hyperparameters of decoding strategies are the same as their reported settings in the original paper or API documentation.

Setting of the Evaluator We feed GPT-4 and GPT-4-Turbo with the prompt shown in Appendix A.5. Both evaluators used the same decoding strategy with top_p=1, max_tokens=10, and frequency_penalty=0. Each evaluator only processes one sub-questions at a time. We calculate the evaluator's average accuracy to represent the generated content's overall quality.

Research Questions To carry out an in-depth investigation of long-form content generation and the effectiveness of our proposed PROXYQA, the following research questions are naturally raised:
RQ1. How do open-sourced LLMs compare to proprietary models' ability to generate extensive reports or articles? (Section 4.2)
RQ2. How well do the modern LLMs grasp the knowledge with different difficulty levels? (Section 4.3)
RQ3. How do LLMs perform in generating long content from different domains? (Section 4.4)
RQ4. How does the iterative and alternate annotation of proxy-questions affect the performance of the LLMs? (Section 4.5)
RQ5. Does PROXYQA give higher ratings for generated content with longer length? (Section 4.6)

4.2 Main Results

The ability of open-sourced LLMs to generate comprehensive and extensive content is far behind the proprietary models. As shown in Table 1, the base versions of the LLaMAs series demonstrate a limited capacity to produce long-form content. However, notable enhancements are achieved through instruction-based supervised fine-tuning (SFT), as the Vicuna-13B and LLaMA2-13B-Chat transcend most other open-sourced models, evidencing their superior capability in delivering acceptable content. However, compared to the proprietary models, the open-source LLMs far lag behind GPT models. Even the GPT-3.5-Turbo outperforms the entire suite of open-source LLMs by a significant margin, and GPT-4-Turbo maintains a substantial lead. This underscores the considerable gap that open-source LLMs must bridge to match the performance of their proprietary counterparts.

4.3 Results on Different Levels of Difficulty

Well-pretrained LLMs surpass others on all fronts but struggle with hard questions, while well-designed retrieval-augmented generation (RAG) significantly make up the shortcoming. Table 1 illustrates a pronounced decline in performance among most large language models (LLMs) on the

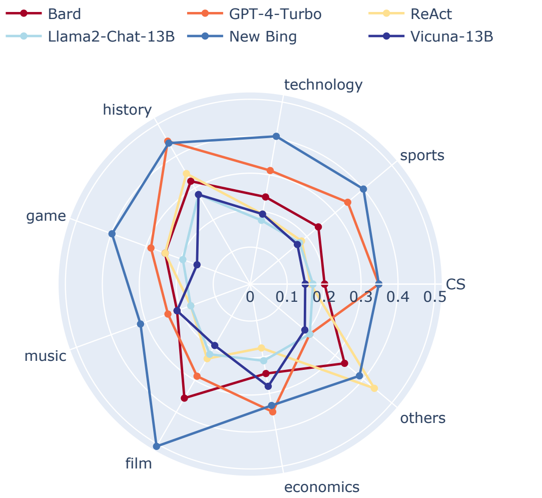

Figure 3: Performance of LLMs on different domains.

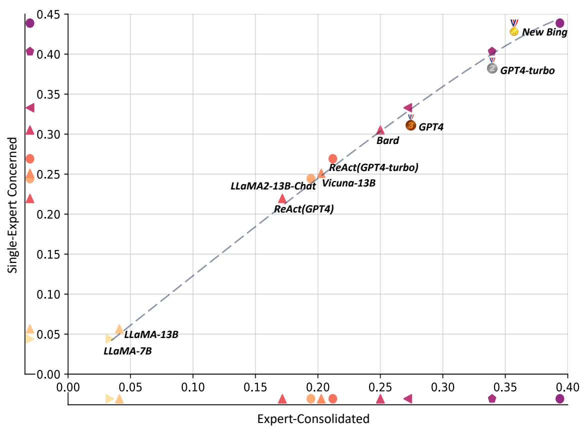

Figure 4: Performance difference on the experts-consolidated and single-expert-focused set.

more challenging subset of questions, which cannot be well-solved solely with information from Wikipedia. Notably, even the powerful GPT-4 exhibits a marked decrease in efficacy(6.55 ↓). In contrast, equipped with the GPT-4 with a search engine and well-designed searching strategy, New Bing Creative Mode performs more robustly than other LLMs, exhibiting a comparatively minor performance loss(39.95 →38.75). However, RAG is not a one-size-fits-all solution, as the GPT-4 model equipped with the ReAct falls short of generating comprehensive content. This is attributed to the fact that ReAct repurposes the GPT into a role more akin to planner and executor, constraining its capacity as a parametric knowledge base.

4.4 Domain

Proprietary LLMs overwhelmingly outperform other competitors in all domains. The advantage is further extended by integrating the search engine. Figure 3 shows that the GPT-4-Turbo surpasses the open-sourced LLMs in all aspects. However, it is worth noting that in some domains, such as music and economics, the gap between open-source models and GPT-4-Turbo is very small, but open-sourced LLMs are biased and inadequate to cover all domains. Training LLMs that excel at multiple domains remains sufficient exploration.

4.5 Impact of Alternate Annotation

Despite the impressive performance, proprietary LLMs are still unable to cater to the preferences of every individual. Figure 4 compares the LLMs' performance on expert-consolidated and single-expert-focused subsets. As the iterative verification and supplementation of proxy-questions proceed, the performance of all models decreases, suggesting that LLMs cannot cater to every individual's preferences. Remarkably, New Bing outperforms all other baselines by a considerable margin, no matter the sub-split where only a single expert is involved or on the complete expert-consolidated set. However, despite the impressive performance, significant degradation could be observed. This suggests that the multi-perspective evaluation criteria in PROXYQA pose critical challenges to LLMs.

4.6 Generation Length

Improving the readability and informativeness of generated content within limited token budgets remains an area for systematic exploration. The average generation lengths are presented in Table 2. It is essential to emphasize that the degree of informativeness and comprehensiveness is not proportional to the length of the generated content. Specifically, LLaMA2-13B generates lengthy content, yet it exhibits the lowest quality in generating contextual background on PROXYQA. In contrast, GPT-4-Turbo produces concise content while conveying extensive and comprehensive information. Moreover, when GPT-4 is incorporated with the search engine, the New Bing Creative Mode yields highly informative and in-depth content, significantly surpassing all other baseline models with an acceptable increase in generation length.

5 Analysis

5.1 Win Rate

We study the pairwise win rate among various LLMs evaluated by PROXYQA and compare the results with human evaluation to validate the effectiveness of PROXYQA.

	Avg. Len.	Acc.
LLaMA2-13B	1906.87	4.11
LLaMA2-13B-Chat	869.42	19.44
Vicuna-13B	727.84	20.26
GPT-3.5-Turbo	823.32	23.94
GPT-4	744.00	27.19
GPT-4-Turbo	1029.47	33.94
ReAct (GPT-4-Turbo)	355.80	21.19
Bard (Gemini Pro)	922.83	25.00
New Bing (Creative)	1167.65	**39.37**

Table 2: Average word count of the generated reports.

Setup Five well-educated postgraduate students are engaged; all have not participated in annotating the meta and proxy-questions of ProxyQA; they are required to score and rank the randomly sampled reports generated by different LLMs. The scoring guideline is shown in Appendix A.4. We sampled ten meta-questions from ProxyQA and employed four LLMs to generate comprehensive reports. As a further comparison, we also follow the settings in MT-Bench (Zheng et al., 2023) that adopt LLM-as-judges, which directly rate the generated report based on the scoring guideline. Similarly, we utilize GPT-Seperate (GPT-S), which evaluates a single report at a time, and GPT-Batch (GPT-B), which evaluates and compares multiple reports simultaneously, to score and rank each report. Given that five human evaluators are involved in the comparison, we ensure fairness and robustness by requiring GPT-S, GPT-B, and our proposed ProxyQA to evaluate each report five times. We then calculate the average win rate based on the pairwise comparison.

Result *GPT-as-judges over-confident on the contents generated by GPT models, while ProxyQA's choice is highly correlated with humans.* As shown in Figure 5, majority of the evaluation results highly correlated with the human's choices. Specifically, evaluators generally recognize the quality of the reports generated by GPT-4-Turbo and New Bing, i.e., their win rates are much higher than those of Vicuna and Llama2-Chat. It is worth noting, however, that GPT-S and GPT-B exhibit overconfidence in the quality of the reports generated by GPT-4-Turbo compared to New Bing. In contrast, both human evaluators and ProxyQA exhibit a preference for New Bing over GPT-4-Turbo. This outcome attests to the effectiveness of ProxyQA and demonstrates the correlation between ProxyQA and human evaluation.

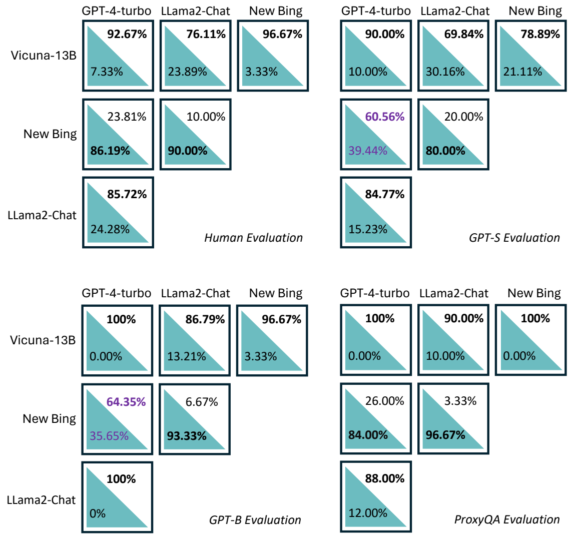

Figure 5: Comparison of win rate of various evaluation methods. GPT-evaluators are highly overconfident in the results produced by GPT-4-Turbo, while ProxyQA significantly correlated with human preference.

5.2 Agreement Evaluation

To thoroughly examine the consistency of ProxyQA and the correlation against human evaluation criteria, we investigate the agreement rate of human assessment and ProxyQA. Two categories of agreement rates are explored. Similar to MT-Bench (Zheng et al., 2023), to effectively gauge the consistency of our proposed ProxyQA, we assess the self-agreement rate, which calculates the inter-evaluator agreement rate. Furthermore, we establish that the evaluation method proposed in the ProxyQA exhibits a strong correlation with human judgment by determining the agreement rate between human evaluations, referred to as the human agreement rate. We employ GPT-Seperate and GPT-Batch, as discussed in section 5.1, for comparison purposes.

Setup Following the settings in section 5.1, 20 reports generated by GPT-4-Turbo and New Bing are evaluated and compared by experts that have not participated in the annotation of ProxyQA. We analyze the agreement between different evaluation methods and human evaluations.

Self-agreement Given a pair of meta-question and its corresponding generated report, each evaluation method is required to score and vote the preferred reports n times. Let $V = \{v_1, \dots, v_n\}$ be the set of voting results, then the self-agreement rate is calculated as:

	Self	G2G	G2M	M2G
GPT-S	48.65	45.49	47.62	30.00
GPT-B	51.17	43.86	36.66	36.66
ProxyQA	**88.00**	**66.00**	**63.33**	**66.19**
Human	52.19	-	-	-

Table 3: Agreement between each evaluation method and human evaluation. Self-agreement is also reported.

$$R_{self} \triangleq \frac{1}{C_n^2}\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbb{1}\{v_i = v_j\} \quad (1)$$

Where $\mathbb{1}\{\cdot\}$ denote the indicator function. The self-agreement rate, denoted as R_{self}, quantifies the consistency of an evaluation method.

Human-agreement The calculation is divided into Majority-to-Group (M2G), Group-to-Majority (G2M) and Group-to-Group (G2G). The R_{M2G} quantifies the proportion of the majority vote of a specific evaluation method in concordance with the overall votes of human evaluation. Conversely, R_{G2M} calculates the proportion of overall votes of an evaluation method that concur with the majority vote of humans, indicating how well the evaluation criteria are aligned with the majority opinion of humans. R_{G2G} provides a view of the overall agreement between an evaluation method and the human. Let $V_e = \{e_1, \ldots, e_n\}, V_h = \{h_1, \ldots, h_n\}$ represent the set of the voting results of an evaluation method and human, respectively. The majority vote of a set is represented as $\mathbb{M}(\cdot)$. Then, the agreement is calculated as:

$$R_{M2G} \triangleq \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\{h_i = \mathbb{M}(V_e)\} \quad (2)$$

$$R_{G2M} \triangleq \frac{1}{n}\sum_{i=1}^{n}\mathbb{1}\{e_i = \mathbb{M}(V_h)\} \quad (3)$$

$$R_{G2G} \triangleq \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{1}\{e_i = h_j\} \quad (4)$$

Results *Human preference varies from individual to individual, while ProxyQA shows firm consistency and is highly correlated to the majority of humans.* Table 3 illustrates that PROXYQA offers a highly consistent metric for evaluating long content generation. Furthermore, PROXYQA strongly correlates with the majority opinion of human evaluators, thereby emphasizing its efficacy in validating long-form content. Notably, the subjective preferences of human experts vary, as the agreement rate reaches only 52.19%, leading to relatively inconsistent evaluations, which is consistent with the findings of Xu et al. (2023). In contrast, PROXYQA achieves an 88.00% agreement rate, indicating its potential as a highly consistent performance indicator. Moreover, when evaluating the consensus between different evaluation methods and the majority opinion of human experts, both GPT-S and GPT-B are contrary to human preference. GPT-as-judges is overconfident in the reports generated by GPT-4-Turbo, which becomes even more extreme when evaluating with GPT-B. However, the criteria of PROXYQA significantly align with the majority opinion of human experts, surpassing GPT-as-judges in all human-agreement rates with a substantial margin and attaining 66.19% and 63.33% M2G and G2M agreement rate respectively. These findings provide robust evidence that the proposed PROXYQA can effectively and reliably assess the capabilities of LLMs in generating long-form content.

5.3 Validation of the GPT Evaluator

We validated the accuracy of our proxy-evaluator. Given the reports generated by GPT-4-Turbo, New Bing, Vicuna-13B, and LLama2-Chat, 100 proxy-questions and the boolean answer generated by the GPT-4 evaluator are sampled. Five human experts involved in the annotation of the proxy-questions are required to validate the evaluator's accuracy and determine whether the answers to the proxy-questions are strictly from the information presented within the generated content. The GPT-4 evaluator reaches a 91% accuracy rate, demonstrating its capability as a reliable evaluator.

6 Conclusion

In this work, we introduce PROXYQA, a framework designed to evaluate LLMs' ability to generate long-form text. Unlike traditional methods that rely on a direct comparison with a reference text, by employing an evaluator to use the information provided in the LLM-generated text to answer proxy-questions, the framework assesses the LLMs' ability without fixed references or crowd-source workers. By mitigating concerns over data contamination and ensuring the relevance and freshness of evaluation content, PROXYQA enhances our understanding of LLMs and drives innovation towards developing long-form generation methods with LLMs.

7 Limitation

In this study, PROXYQA mainly focuses on evaluating the informative and knowledge coverage of the generated texts. However, multiple key dimensions such as factuality (Min et al., 2023), verifiability (Liu et al., 2023), and coherency (Deng et al., 2022) should be considered for long-form content generation. For instance, proxy questions cannot measure hallucination in long-form content, a critical issue for a long-form generation. On the other hand, each meta-question is annotated by five experts, but they cannot cover potentially all the proxy-questions. More advanced methods that consider these issues will be developed in future work.

Ethical Considerations

To avoid potential ethical issues, we carefully checked all questions in multiple aspects, as discussed in Section 3.3. We try to guarantee that all samples do not involve any offensive, gender-biased, or political content, and any other ethical issues. The source code will be released with instructions to support correct use.

References

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *CoRR*, abs/2307.11088.

Anthropic. 2023. Claude 2.1 with 200K Context Window.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *CoRR*, abs/2308.14508.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

Bing. 2023. Ai-powered bing with chatgpt's gpt-4.

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

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *CoRR*, abs/2006.14799.

Harrison Chase. 2022. Langchain.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *CoRR*, abs/2306.15595.

David Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 15607–15631. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *CoRR*, abs/2307.08691.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *NeurIPS*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4599–4610. Association for Computational Linguistics.

Yuntian Deng, Volodymyr Kuleshov, and Alexander M. Rush. 2022. Model criticism for long-form text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11887–11912. Association for Computational Linguistics.

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2023. BAMBOO: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *CoRR*, abs/2309.13345.

Dom Eccleston. 2022. Sharegpt.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019a. ELI5: long form question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3558–3567. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2019b. Strategies for structuring story generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2650–2660. Association for Computational Linguistics.

Gemini. 2023. Gemini: A family of highly capable multimodal models.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Dan Gillick and Yang Liu. 2010. Non-expert evaluation of summarization systems is risky. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 148–151, Los Angeles. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. SNaC: Coherence error detection for narrative summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 444–463, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhe Hu, Hou Pong Chan, Jiachen Liu, Xinyan Xiao, Hua Wu, and Lifu Huang. 2022. PLANET: dynamic content planning in autoregressive transformers for long-form text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2288–2305. Association for Computational Linguistics.

Xinyu Hua and Lu Wang. 2020. PAIR: planning and iterative refinement in pre-trained transformers for long text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 781–793. Association for Computational Linguistics.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Mia Xu Chen, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. 2019. *GPipe: Efficient Training of Giant Neural Networks Using Pipeline Parallelism*. Curran Associates Inc., Red Hook, NY, USA.

Neslihan Iskender, Tim Polzehl, and Sebastian Mller. 2020. Best practices for crowd-based evaluation of German summarization: Comparing crowd, expert and automatic evaluation. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 164–175, Online. Association for Computational Linguistics.

Yuchen Jiang, Tianyu Liu, Shuming Ma, Dongdong Zhang, Jian Yang, Haoyang Huang, Rico Sennrich, Ryan Cotterell, Mrinmaya Sachan, and Ming Zhou. 2022. BlonDe: An automatic evaluation metric for document-level machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1550–1565, Seattle, United States. Association for Computational Linguistics.

Dongyeop Kang and Eduard H. Hovy. 2020. Plan ahead: Self-supervised text planning for paragraph completion task. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6533–6543. Association for Computational Linguistics.

Ashkan Kazemi, Kiran Garimella, Devin Gaffney, and Scott Hale. 2021. Claim matching beyond English to scale global fact-checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4504–4517, Online. Association for Computational Linguistics.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Yoonjoo Lee, Kyungjae Lee, Sunghyun Park, Dasol Hwang, Jaehyeon Kim, Hong-In Lee, and Moontae

Lee. 2023. QASA: advanced question answering on scientific articles. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 19036–19052. PMLR.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2511–2522. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 12076–12100. Association for Computational Linguistics.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *CoRR*, abs/2305.16300.

Reiichiro Nakano, Jacob Hilton, S. Arun Balaji, Jeff Wu, Ouyang Long, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback. *ArXiv*, abs/2112.09332.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

OpenAI. 2022a. ChatGPT.

OpenAI. 2022b. Text-davinci-003.

OpenAI. 2023a. GPT-3.5 Turbo.

OpenAI. 2023b. GPT-4 Technical Report. *CoRR*, abs/2303.08774.

OpenAI. 2023c. GPT-4 Turbo with 128K Context.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

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

Justus Randolph. 2010. Free-marginal multirater kappa (multirater kfree): An alternative to fleiss fixed-marginal multirater kappa. volume 4.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud D. Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2022. Parallel context windows for large language models. In *Annual Meeting of the Association for Computational Linguistics*.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 10776–10787. Association for Computational Linguistics.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *CoRR*, abs/2305.14196.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and Omer Levy. 2022. SCROLLS: standardized comparison over long language sequences. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 12007–12021. Association for Computational Linguistics.

Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. 2023. Large language models are not yet human-level evaluators for abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 4215–4233. Association for Computational Linguistics.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8273–8288. Association for Computational Linguistics.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *ArXiv*, abs/2104.09864.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

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, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Szymon Tworkowski, Konrad Staniszewski, Mikolaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Milos. 2023. Focused transformer: Contrastive training for context scaling. *CoRR*, abs/2307.03170.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A critical evaluation of evaluations for long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 3225–3245. Association for Computational Linguistics.

Jing Xu, Arthur Szlam, and Jason Weston. 2022. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5180–5197. Association for Computational Linguistics.

Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. MEGATRON-CNTRL: controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2831–2845. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023a. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Qihao Wu, Junhao Chen, Zihang Xu, Zhenning Dai, Xu Han, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2023. Infinitebench: 128k long-context benchmark for language models.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Appendix

A.1 Advancements in Long-Context LLMs

As LLMs gain global traction, their ability to adeptly process extensive sequences—such as protracted conversation histories or intricate scientific documents—becomes increasingly significant. Recent years have witnessed the emergence of LLMs capable of managing long context windows, a feat made possible by the advent of more powerful GPUs with expanded memory capacities, along with innovations in memory-efficient attention mechanisms (Dao et al., 2022; Dao, 2023).

Historically, the context window size for models like GPT-2 encompassed 1024 tokens (Radford et al., 2019), which was then extended to 2048 in GPT-3 (Brown et al., 2020). Modern iterations, such as GPT-4-turbo, boast an impressive 128K token capacity (OpenAI, 2023c), while Claude 2.1 extends this even further to 200K tokens (Anthropic, 2023). Nevertheless, scaling the context window during the pretraining phase remains a daunting task as the computational demands surge quadratically with the length of the attention span, and a majority of texts within standard corpora, like Common Crawl, tend to be comparatively brief.

A novel approach gaining momentum among researchers is the augmentation of the LLMs' context window through the process of continued training or fine-tuning. For instance, Tworkowski et al. (2023) successfully refined the 3B and 7B OpenLLaMA checkpoints, employing contrastive training techniques to adeptly handle contexts stretching up to 8K tokens. Similarly, Mohtashami and Jaggi (2023) achieved an expansion of the context length from 4K to 32K for LLaMA 7B by incorporating "landmark tokens" that effectively encapsulate blocks of the existing context. These tokens allow for focused fine-tuning of attention mechanisms, which in turn facilitates the selection of pertinent contextual blocks.

Furthermore, Chen et al. (2023) introduced a method known as positional interpolation, to be used with LLMs that incorporate Rotary Position Embeddings (RoPE) as their choice of positional encoding (Su et al., 2021). This technique yielded promising outcomes when applied to LLaMA models ranging from 7B to 65B in size, requiring minimal fine-tuning efforts—a mere 1000 optimization steps. A different paradigm, ALiBi (Press et al., 2022), circumvents the necessity of fine-tuning altogether for expanding the context window. By eschewing positional embeddings and instead applying a linear bias to the attention scores—which is proportionate to the distance between tokens—it elegantly adjusts to handle longer contexts.

Lastly, the strategy proposed by Ratner et al. (2022) partitions extensive contexts into several sub-windows, employing the same positional embeddings across them. This innovative reuse of embeddings enables the models to cope with extended contexts without the need for additional fine-tuning. This collective body of work represents the ongoing evolution of strategies to enhance the capabilities of LLMs in accommodating long context sequences, a critical requirement for their effective deployment in complex, real-world applications.

A.2 Evaluation for Long-Context LLMs

The advent of long-context LLMs has ushered in an era where evaluating performance over extensive text sequences is crucial. Benchmarks like ZeroSCROLLS (Shaham et al., 2023) have emerged to challenge these models' understanding of expansive texts in a zero-shot setting. ZeroSCROLLS extends the foundation laid by the SCROLLS benchmark (Shaham et al., 2022)—originally designed to handle longer texts through fine-tuning—by introducing four new tasks: query-based summarization, multi-hop question answering, sentiment aggregation, and ordering book chapter summaries. It distinguishes itself by focusing on zero-shot performance, using simple natural language prompts and eschewing training data, relying on non-public, high-quality references.

Another contribution to this domain is LongBench (Bai et al., 2023), a suite of 21 datasets across 6 categories of tasks such as single- and multi-document question answering, summarization, few-shot learning, specific synthetic tasks, and code completion. What sets LongBench apart is its uniform format for all datasets, promoting a unified and automated evaluation process with metrics like F1 and ROUGE. Bamboo (Dong et al., 2023) also provides a valuable framework for analyzing comprehension over lengthy texts, offering a selection of 10 datasets from 5 diverse activities that range from question answering to hallucination detection, text sorting, language modeling, and code completion. Bamboo specifically tackles potential data contamination by exclusively using sources released no earlier than 2023, maintaining the relevance and contemporaneity of its material.

L-Eval (An et al., 2023) introduces a bifurcated approach to evaluate LLMs, featuring both closed-ended and open-ended tasks. Closed-ended tasks focus on the model's reasoning and comprehension skills in a protracted context. In contrast, its open-ended tasks provide a variety of summarization challenges that require models to synthesize information from lengthier documents. InfiniteBench (Zhang et al., 2023) is tailored to assess LLMs that process, understand, and infer information from contexts that span over 100,000 tokens. It should be noted that these datasets prioritize the assessment of long-context understanding, and as a result, a significant portion of tokens are used as inputs for the LLMs rather than outputs.

A.3 Baseline Models

All the baselines are prompted with "Write a well-structured and extensive report to answer the question: [META QUESTION]". We then employ the generated results as the contextual background and force the GPT-4 and GPT-4-turbo to answer the proxy-questions accordingly. Multiple competitive baselines are tested under PROXYQA.

Base LLaMA is a set of open-sourced LLMs pretrained on diverse sources spanning multiple domains (Touvron et al., 2023a,b). The pretraining corpora include the June 2022 Wikipedia dumps, which should enable the LLaMA family to effectively address most 'easy' meta-questions in PROXYQA. Our experiment evaluates LLaMA-7B, LLaMA2-7B, and LLaMA2-13B.

Instruction-Finetuned LLaMA includes Vicuna (Chiang et al., 2023), Alpaca (Taori et al., 2023), LLaMA2-Chat (Touvron et al., 2023b). Vicuna is a chat assistant trained by fine-tuning LLaMA on around 70k user-shared conversations collected from ShareGPT (Eccleston, 2022). Similarly, Alpaca is trained with 52k self-instructed demonstrations adapted text-davinci-003 (OpenAI, 2022b). As an extension of base LLaMA2, LLaMA2-Chat is optimized specifically for dialogue usage of over 1 million instructions.

OpenAI APIs includes GPT-3.5-turbo(OpenAI, 2023a), GPT-4 and GPT-4-turbo (OpenAI, 2023b). The default decoding configuration is utilized to generate responses, while the maximum decoding length is set as its maximum limitation. Both GPT-3.5-turbo and GPT-4-turbo are of version 1106, while GPT-4 corresponds to GPT-4-0613. The training data for GPT-4-turbo is up-to-date as of April 2023, while the remaining models are trained with data up to September 2021.

Web-Augmented LLMs utilize external search APIs are evaluated. Specifically, GPT-4 and GPT-4-turbo are integrated with the Google Search API under the configuration of ReAct (Yao et al., 2023b). These models are tasked with processing meta-questions, reasoning through search traces, and extracting relevant content from search results from the internet across multiple turns. The implementation is adopted from LangChain (Chase, 2022). In addition to ReAct, the performance of New Bing (creative mode) (Bing, 2023) and Bard with Gemini Pro (Gemini, 2023) are also assessed.

A.4 Scoring Guideline for Human Evaluation and LLM-as-Judges

Scoring Guideline

Please rate the knowledge coverage of the reports provided, using a scale of 0-5. Assess how well the report covers the necessary information related to the question.
Knowledge Coverage Scale:
0 - Nonsense: The report offers no useful information and is completely irrelevant to the question.
1 - Poor: The report provides very little useful information and barely addresses the question.
2 - Fair: The report offers some useful information but lacks depth and detail, leaving the question partially unanswered.
3 - Average: The report presents a decent amount of information, addressing the question adequately but not exceptionally.
4 - Good: The report provides comprehensive information, covering the question well with appropriate depth and detail.
5 - Excellent: The report thoroughly covers all aspects of the question, offering a high level of detail and leaving no gaps in knowledge.

A.5 Prompts for tested LLMs and Evaluator

All the tested baselines are prompted with "Write a well-structured and extensive report to answer the question: [META QUESTION]", while the evaluators are prompted with:

Prompt for Evaluator

Read the provided document and determine whether the statement below is "True" or "False". Use only the information in the text to make your decision. Do not rely on prior knowledge or information outside of the given text. If the text does not provide enough information to make a decision, respond with "Not mentioned".
Format your answer as "True", "False", or "Not mentioned".
Document: [generated_report]
Statement: [proxy_question]

A.6 Annotation Guideline for Formulating Meta-Questions and Proxy-questions

The meta-questions should be based on the topic the experts are most familiar with or keen on.
Meta-question: The meta-question should be:
a) Answerable through thorough research. b) Aligned with real-life scenarios. c) Avoid of offensive or ethical concerns. d) Without an absolute or unique answer. e) Addressable in long-form reports or articles. f) Ensure the questions are open-ended, promoting in-depth research and discussion.
Proxy-questions: For each meta-question, determine the essential proxy-questions that cover the critical contents of the topic. proxy-questions should be:
a) Directly related to the meta-question. b) Comprehensive enough to cover different angles of the meta-question.
Golden Answer: For each proxy-question, label the golden answer, which refers to the most accurate and relevant information.
Steps for Annotation:
a) Choose a topic you are familiar with or interested in. b) Formulate a meta-question following the criteria mentioned above c) Determine the essential proxy-questions that cover the critical contents of the topic, as per the guidelines above. d) For each proxy-question, label the golden answer. e) Review the meta-question, proxy-questions, and golden answer to ensure they adhere to the guidelines and criteria provided. f) Collaborate with your peers to review and refine the meta-questions and proxy-questions. Verify and supplement the proxy-questions in an alternate way.

A.7 Case Study and Error Analysis

Table 4- 9 show the case study of the reports generated by LLaMA2-13B-Chat, New Bing and GPT-4-turbo. Although LLaMA2-13B-Chat generates lengthy contents, very little important information are contained. Therefore GPT-4 evaluator is unable to fetch useful information from the generated contents, result in low accuracy on the proxy-questions. In contrast, both New Bing and GPT-4-turbo generate more concise results conveyed with essential

contents. Specifically, critical hallucination can be observed in the output of the LLaMA2-13B-Chat, while none of the contents can be used to answer the proxy-questions. However, New Bing covers as much points as it can, providing a comprehensive overview of the given topic, while results of GPT-4-turbo is well-structured and offers detailed and helpful introduction to some of the critical contents.

A.8 Distribution of the ProxyQA

51% of meta-questions are categorized as easy, signifying that they can be addressed utilizing information solely from Wikipedia. In contrast, 49% of meta-questions are deemed difficult. Moreover, 53% of these questions are time-sensitive, as their answers may dynamically alter over time. A representative example of a time-sensitive question is: "Could you kindly offer a comprehensive overview of the historical development of pre-training in NLP, encompassing notable contributions?"

A.9 Maintaince of the Leaderboard

Concerning meta-questions of varying difficulty levels, the average accuracy of evaluators will be reported for both easy and hard subsets of PROXYQA. In addition, for time-sensitive questions, proxy-questions will be incrementally updated, and the leaderboard will be dynamically adjusted to evaluate the LLMs' possession of current knowledge. The leaderboard is updated on a weekly basis, while the proxy-question of time-sensitive meta-question is incrementally updated on a bi-monthly basis. Each periodic update of proxy-questions will also renew the leaderboard.

Question: Data parallelism, model parallelism, and pipeline parallelism play a vital role in the training of large-scale language models. What are the representative works and frameworks among these technologies? Please introduce these technologies and frameworks in detail.

LLaMA2-13B-Chat-Page 1
Introduction: The training of large-scale language models has become increasingly popular in recent years due to their wide range of applications in natural language processing (NLP) tasks. However, training such models can be computationally expensive and time-consuming, especially when dealing with large datasets. To address this challenge, researchers have proposed various parallelization techniques to speed up the training process. In this report, we will focus on three key parallelization techniques: data parallelism, model parallelism, and pipeline parallelism. We will introduce representative works and frameworks that utilize these techniques and discuss their advantages and limitations. **Data Parallelism:** Data parallelism is a technique that involves dividing the input data into smaller chunks and processing them in parallel across multiple devices or machines. This technique is particularly useful for training large-scale language models, as it allows for faster training times and more efficient use of computational resources. One of the most popular frameworks for data parallelism is TensorFlow, which is an open-source machine learning library developed by Google. TensorFlow provides a built-in data parallelism feature called TensorFlow Estimator, which allows users to parallelize their training processes across multiple machines. Another popular framework for data parallelism is PyTorch, which provides a module called PyTorch Distributed that allows users to parallelize their training processes across multiple devices. **Representative Works:** One of the most influential works in data parallelism for large-scale language models is the paper "Distributed Training of Deep Neural Networks" by Google researchers (2016). This paper introduced the concept of using data parallelism to train deep neural networks on large datasets, and demonstrated the effectiveness of this approach for training large-scale language models. Another influential work is the paper "Deep Learning with CUDA: General Purpose Training of Neural Networks on GPU Clusters" by NVIDIA researchers (2017). This paper introduced the concept of using GPU clusters for training large-scale language models, and demonstrated the effectiveness of this approach for improving training times and reducing computational costs. **Model Parallelism:** Model parallelism is a technique that involves dividing the model into smaller parts and processing them in parallel across multiple devices or machines. This technique is particularly useful for training large-scale language models, as it allows for faster training times and more efficient use of computational resources. One of the most popular frameworks for model parallelism is the open-source machine learning library TensorFlow, which provides a built-in feature called TensorFlow Model Parallelism. This feature allows users to parallelize their training processes across multiple machines, and has been used in a number of influential works for training large-scale language models. Another popular framework for model parallelism is the PyTorch framework, which provides a module called PyTorch Model Parallel that allows users to parallelize their training processes across multiple devices.

Table 4: Case study of LLaMA2-13B-Chat

LLaMA2-13B-Chat-Page 2
Representative Works: One of the most influential works in model parallelism for large-scale language models is the paper "Model Parallelism for Training Large-Scale Neural Networks" by Google researchers (2017). This paper introduced the concept of using model parallelism to train large-scale language models, and demonstrated the effectiveness of this approach for improving training times and reducing computational costs. **Pipeline Parallelism:** Another influential work is the paper "Scaling Up Neural Machine Translation with Model Parallelism" by Google researchers (2018). This paper introduced the concept of using model parallelism to train large-scale neural machine translation models, and demonstrated the effectiveness of this approach for improving translation quality and reducing computational costs. Pipeline parallelism is a technique that involves dividing the training process into smaller stages and processing them in parallel across multiple devices or machines. This technique is particularly useful for training large-scale language models, as it allows for faster training times and more efficient use of computational resources. One of the most popular frameworks for pipeline parallelism is the open-source machine learning library TensorFlow, which provides a built-in feature called TensorFlow Pipeline Parallelism. This feature allows users to parallelize their training processes across multiple machines, and has been used in a number of influential works for training large-scale language models. Another popular framework for pipeline parallelism is the PyTorch framework, which provides a module called PyTorch Pipeline that allows users to parallelize their training processes across multiple devices. **Representative Works:** One of the most influential works in pipeline parallelism for large-scale language models is the paper "Pipeline Parallelism for Training Large-Scale Neural Networks" by Google researchers (2018). This paper introduced the concept of using pipeline parallelism to train large-scale language models, and demonstrated the effectiveness of this approach for improving training times and reducing computational costs. Another influential work is the paper "Scaling Up Neural Machine Translation with Pipeline Parallelism" by Google researchers (2019). This paper introduced the concept of using pipeline parallelism to train large-scale neural machine translation models, and demonstrated the effectiveness of this approach for improving translation quality and reducing computational costs. **Conclusion:** In conclusion, data parallelism, model parallelism, and pipeline parallelism are three key parallelization techniques that have played a vital role in the training of large-scale language models. These techniques have been used in a number of influential works, and have demonstrated the effectiveness of parallelizing the training process to improve training times and reduce computational costs. As the field of NLP continues to evolve, it is likely that these parallelization techniques will continue to play an important role in the training of large-scale language models.

Table 5: Case study of LLaMA2-13B-Chat

New Bing - Page 1

Hello, this is Bing. I can help you with your question.
Data parallelism, model parallelism, and pipeline parallelism are three types of parallelism methods that can be used to train large-scale language models efficiently on multiple GPUs or other devices. Here is a brief introduction of these technologies and some representative works and frameworks that use them:

- **Data parallelism (DP)**: This is the simplest and most common form of parallelism, where the same model is replicated on multiple devices, and each device processes a different slice of the input data. The gradients are then aggregated and synchronized across the devices to update the model parameters. This method is easy to implement and can scale well with the number of devices, but it has some limitations, such as requiring large memory to store the full model and communication overhead for gradient synchronization. Some examples of frameworks that support data parallelism are PyTorch's **DistributedDataParallel** and TensorFlow's **MirroredStrategy**.
- **Model parallelism (MP)**: This is a more complex form of parallelism, where the model is split into smaller parts and distributed across multiple devices, such that each device only stores and computes a subset of the model parameters. This method can overcome the memory limitation of data parallelism and enable training very large models that cannot fit on a single device. However, it also introduces more challenges, such as how to partition the model, how to minimize the communication cost, and how to balance the workload among the devices. There are different types of model parallelism, such as tensor parallelism and pipeline parallelism, which will be explained below. Some examples of frameworks that support model parallelism are NVIDIA's **Megatron-LM** and Microsoft's **DeepSpeed**.
- **Tensor parallelism (TP)**: This is a type of model parallelism where each tensor (such as a weight matrix or an activation vector) is split into smaller chunks along one or more dimensions, and each chunk is assigned to a different device. For example, a weight matrix of size [m, n] can be split into [m/k, n/k] chunks, where k is the number of devices, and each device only stores and computes one chunk. This method can reduce the memory footprint and communication cost by a factor of k, but it also requires modifying the model architecture and the computation logic to handle the tensor sharding. Some examples of works that use tensor parallelism are **gShard** and **Mesh-TensorFlow**.
- **Pipeline parallelism (PP)**: This is another type of model parallelism where the model is split into smaller stages (such as layers or groups of layers) and distributed across multiple devices, such that each device only stores and computes one stage of the model. For example, a model with L layers can be split into S stages, where each stage contains L/S layers, and each device only processes one stage. The input data is then fed into the first device, and the output of each device is passed to the next device in a pipelined fashion. This method can also reduce the memory footprint and communication cost by a factor of S, but it also requires careful scheduling and synchronization of the pipeline stages to maximize the throughput and avoid idle periods. Some examples of works that use pipeline parallelism are **PipeDream**, **GPipe**, and **PipeEdge**.

Table 6: Case study of New Bing

New Bing - Page 2
These parallelism methods can be combined in different ways to achieve higher efficiency and scalability. For example, one can use data parallelism and tensor parallelism together to train a large model on a large number of devices, as done in **Megatron-LM**. Alternatively, one can use data parallelism and pipeline parallelism together to train a deep model with high throughput, as done in **GPipe**. Furthermore, one can use tensor parallelism and pipeline parallelism together to train a very large and deep model with low memory and communication cost, as done in **DeepSpeed**. I hope this helps you understand the basics of parallelism methods for large-scale language model training. If you want to learn more, you can check out the links I provided or search for more resources on Bing. Have a nice day! (1) Model Parallelism - Hugging Face. https://huggingface.co/docs/transformers/v4.15.0/parallelism. (2) undefined. https://github.com/nvidia/megatron-lm. (3) Efficient Large-Scale Language Model Training on GPU Clusters - arXiv.org. https://arxiv.org/pdf/2104.04473v2. (4) PipeEdge: Pipeline Parallelism for Large-Scale Model Inference on .... https://ieeexplore.ieee.org/document/9996638. (5) Efficient Large-Scale Language Model Training on GPU Clusters - arXiv.org. https://arxiv.org/pdf/2104.04473v2. (6) Optimizing Distributed Training on Frontier for Large Language Models. https://arxiv.org/html/2312.12705v2. (7) Abstract arXiv:2102.07988v2 [cs.LG] 28 Sep 2021. https://arxiv.org/pdf/2102.07988.pdf.

Table 7: Case study of New Bing

GPT-4-turbo - Page 1

A.10 Introduction

The training of large-scale language models, such as GPT-3, BERT, and T5, requires significant computational resources due to the vast number of parameters and the size of the datasets involved. To address this challenge, parallelism techniques have been developed to distribute the computational workload across multiple processors or machines. The three main types of parallelism used in training large-scale language models are data parallelism, model parallelism, and pipeline parallelism. This report provides an in-depth look at these technologies and their representative works and frameworks.

A.11 Data Parallelism

A.11.1 Definition and Overview

Data parallelism involves splitting the training data across multiple processors or devices, with each processor holding a complete copy of the model. Each processor computes the gradients based on its subset of the data, and the gradients are then aggregated to update the model parameters.

A.11.2 Representative Works and Frameworks

- **Horovod**: Developed by Uber, Horovod is a distributed training framework that uses data parallelism. It is designed to work with TensorFlow, Keras, PyTorch, and Apache MXNet. Horovod simplifies the process of scaling out deep learning models by providing efficient gradient aggregation.
- **PyTorch Distributed Data Parallel (DDP)**: PyTorch's DDP module allows for easy data parallelism implementation. It wraps the model during training and manages gradient averaging and synchronization across multiple GPUs.
- **TensorFlow's `tf.distribute.Strategy`**: TensorFlow offers several strategies for distributed training, including `MirroredStrategy` for data parallelism. It synchronizes training across multiple GPUs by replicating the model on each device and managing gradient updates.

A.12 Model Parallelism

A.12.1 Definition and Overview

Model parallelism involves splitting the model itself across multiple processors or devices. Each processor is responsible for a portion of the model's layers or parameters. This technique is particularly useful for models that are too large to fit into the memory of a single device.

A.12.2 Representative Works and Frameworks

- **GPipe**: Developed by Google, GPipe is a library for efficiently training large-scale neural networks using model parallelism. It partitions the model into different segments and pipelines the execution across multiple accelerators.
- **Megatron-LM**: Created by NVIDIA, Megatron-LM is a framework designed to train transformer-based models using model parallelism. It splits the model's layers across multiple GPUs, allowing for the training of models with billions of parameters.
- **Mesh-TensorFlow**: This is a language for distributed deep learning, developed by Google, that allows for easy implementation of model parallelism. It provides a simple API to distribute the model's computation across a mesh of processors.

Table 8: Case study of GPT-4-turbo

GPT-4-turbo - Page 2

A.13 Pipeline Parallelism

A.13.1 Definition and Overview

Pipeline parallelism combines elements of both data and model parallelism. The model is split across multiple processors, and the training data is also split into micro-batches. Each processor works on a different micro-batch and passes its output to the next processor in the pipeline.

A.13.2 Representative Works and Frameworks

- **GPipe**: While GPipe is primarily associated with model parallelism, it also incorporates aspects of pipeline parallelism by pipelining the execution of micro-batches across different model partitions.
- **DeepSpeed**: Developed by Microsoft, DeepSpeed is a deep learning optimization library that provides a pipeline parallelism implementation called ZeRO-DP. It is designed to optimize memory usage and improve scalability when training large models.
- **FairScale**: An extension of PyTorch, FairScale includes several features for distributed training, including pipeline parallelism. It allows for the efficient training of large models by partitioning the workload across multiple GPUs.

A.14 Conclusion

Data parallelism, model parallelism, and pipeline parallelism are essential techniques for training large-scale language models. Each technique addresses different challenges associated with the size of the model and the dataset. Frameworks like Horovod, PyTorch DDP, TensorFlow's `tf.distribute.Strategy`, GPipe, Megatron-LM, Mesh-TensorFlow, DeepSpeed, and FairScale provide the necessary tools for implementing these parallelism strategies effectively.

As the demand for larger and more complex models continues to grow, these parallelism techniques and frameworks will play a crucial role in enabling the development of next-generation AI systems. Researchers and practitioners must carefully choose the appropriate parallelism strategy based on the specific requirements of their models and the available computational resources. With ongoing advancements in distributed computing and parallel processing, we can expect to see further improvements in the efficiency and scalability of large-scale language model training.

Table 9: Case study of GPT-4-turbo