Tree-Instruct: A Preliminary Study of the Intrinsic Relationship between Complexity and Alignment

Yingxiu Zhao[1], **Bowen Yu**[2], **Binyuan Hui**[2], **Haiyang Yu**[2], **Minghao Li**[2], **Fei Huang**[2], **Nevin L. Zhang**[1], **Yongbin Li**[2]
The Hong Kong University of Science and Technology, Alibaba Group
{yzhaocx,lzhang}@connect.ust.hk
{yubowen.ybw,binyuan,yifei,liminghao,f.huang,shuide}@alibaba-inc.com

Abstract

Training large language models (LLMs) with open-domain instruction data has yielded remarkable success in aligning to end tasks and human preferences. Extensive research has highlighted the importance of the quality and diversity of instruction data. However, the impact of data complexity, as a crucial metric, remains relatively unexplored from three aspects: (1)where the sustainability of performance improvements with increasing complexity is uncertain; (2)whether the improvement brought by complexity merely comes from introducing more training tokens; and (3)where the potential benefits of incorporating instructions from easy to difficult are not yet fully understood. In this paper, we propose *Tree-Instruct* to systematically enhance the instruction complexity in a controllable manner. By adding a specified number of nodes to instructions' semantic trees, this approach not only yields new instruction data from the modified tree but also allows us to control the difficulty level of modified instructions. Our preliminary experiments reveal the following insights: (1)Increasing complexity consistently leads to sustained performance improvements of LLMs. (2)Under the same token budget, a few complex instructions outperform diverse yet simple instructions. (3)Curriculum instruction tuning might not yield the anticipated results; focusing on increasing complexity appears to be the key. Code and data are released: https://github.com/xiuzbl/Tree-Instruct.

Keywords: controllable instruction evolution, instruction complexity, large language models

1. Introduction

The latest generation of large language models (LLMs) has attracted significant attention due to their immense potential in language technologies (OpenAI, 2022; Touvron et al., 2023; Wei et al., 2023; Li et al., 2023). To enhance interactive user requests and chat interfaces, these models undergo instruction-tuning using supervised input-output pairs (Iyer et al., 2022; Jang et al., 2023; Chung et al., 2022). This process enables the model to comprehend the required style and format for effective user interaction, showcasing the knowledge and capabilities gained during pretraining (Ouyang et al., 2022).

Consequently, the efficacy of instruction data significantly influences LLMs' abilities, shaping users' perceptions of their capabilities (Wang et al., 2023g; Köpf et al., 2023; Chiang et al., 2023). Recently, LIMA has demonstrated that with just 1000 carefully curated prompts and responses, an LLM can achieve remarkably strong performance (Zhou et al., 2023). This suggests that the scaling laws of instruction tuning are not solely dependent on data quantity but rather influenced by prompt diversity and quality. However, one critical and less-explored aspect of evaluating instruction data is complexity. There are at least three unanswered questions related to complexity: (1) ***The scaling law of instruction complexity****: Intuitively, more complex instruction data might elicit more potential capabilities in LLMs to address intricate problems (Luo et al., 2023; Mukherjee et al., 2023)*. WizardLM (Xu et al., 2023) introduce in-depth and in-breadth evolving methods to rewrite prompts into more complex and diverse versions, resulting in a 12.4% increase in LLMs' win rate with the same amount of data. Yet, whether WizardLM's performance improvement is due to complexity or merely derived from diversity remains uncertain. Moreover, the ongoing enhancements in complexity are yet to be explored. (2) ***The relationship between complexity-induced performance improvement and token quantity***: Enhancing instance complexity inevitably increases the number of tokens per instance (Dai et al., 2021). While WizardLM exhibits performance improvements with the same instance quantity, it increases the number of tokens per instance. This raises the question of whether complexity-induced improvement in LLMs results from increased training tokens. As known, enlarging LLMs' pretraining token counts can lead to better performance (Muennighoff et al., 2023; Tay et al., 2022). (3) ***The effectiveness of complexity-based curriculum instruction learning***: Curriculum learning is a strategy in machine learning that starts with easy instances and gradually introduces harder ones (Bengio et al., 2009). Its effectiveness has been demonstrated in various NLP tasks like machine translation (Zhou et al., 2020), dialogue systems (Zhu et al., 2021), and question answer-

Correspondence to: Bowen Yu, Yongbin Li.

LREC-COLING 2024, pages 16776–16789

ing (Sachan and Xing, 2016). However, its potential efficacy in instruction tuning is still under-explored.

However, to answer the aforementioned questions, the key hurdle lies in finding a controlled way to increase the complexity of instruction data without introducing unwanted factors such as diversity. WizardLM (Xu et al., 2023) employs an in-depth evolving prompt like "*Your objective is to rewrite a given prompt into a more complex version to make ChatGPT and GPT4 a bit harder to handle.*" to complicate the existing instructions. Unfortunately, although intended to enhance complexity, this approach might inadvertently introduce diversity by diverting from the initial instruction objectives. This issue becomes particularly severe when repeatedly employing in-depth evolving to achieve varying levels of complexity. We study and analyze the instructions before and after in-depth evolving in Sec. 4.1. As illustrated in Fig. 2, the iteratively evolved instructions append additional objectives that deviate from the original instructions, showcasing a greater diversity.

To address this concern, we propose ***Tree-Instruct***, which involves prompting LLMs to add a specific number of new nodes to the semantic tree of an existing instruction, as opposed to manipulating the text sequence directly, as done in Self-Instruct (Wang et al., 2022a) or WizardLM (Xu et al., 2023). We use the number of added nodes to represent the introduced level of complexity. The advantage of this approach lies in the fact that semantic tree nodes lack any sequential order (Shiv and Quirk, 2019). By enforcing LLMs to operate on the semantic tree, this process becomes analogous to inserting new words into the middle of the original instructions. This compels the models to complicate while adhering to the structural constraints of the initial instruction rather than merely appending new instructions. It can significantly mitigate the issue of straying from the primary theme of the initial instruction. We leverage GPT-4 to assess the consistency of evolved instructions with original ones, and the results verify that Tree-Instruct improves WizardLM's consistency score from 0.56 to 0.69. Fig. 1 highlights how the number of added nodes raises the complexity level of the samples.

With the help of Tree-Instruct, we have obtained the following preliminary experimental conclusions:

(1) ***As the complexity of the instruction data increases, the benefits of instruction tuning continue to grow***: Following LIMA(Zhou et al., 2023), we attempt instruction tuning using 1,000 samples from Alpaca-GPT-4 as a base. We add 3, 6, and 10 nodes to the semantic tree of each sample, resulting in performance gains of 13%, 20%, and 26%, respectively, across eight sub-skills such as commonsense, writing, and coding, showing consistent improvements. Furthermore, this scaling law can be extended to more complex instruction data. For instance, when fine-tuning around 6,000 conversations filtered from ShareGPT via OpenChat(Wang et al., 2023a) (showing excellent performance in the open-source LLMs), we observe that by increasing the complexity through Tree-Instruct to around 1,100 users' instructions, the winning rate increases from 84.56% to 86.19% for AlpacaEval benchmark[1].

(2) ***The increase in complexity partly comes from additional tokens, but a few complex instructions outperform diverse yet simple instructions, under the same token budget.***: We find that as the complexity increases, the number of tokens also increases. Adding ten nodes in the semantic tree increases the average token length of instructions from 186 to 607. Hence, to make a fair comparison, we increase the number of original instructions from 1,000 to 4,000 to match the total token quantity of our tree-instructed samples. Under this setting, the performance gain from adding ten nodes still achieves more than 15%. This indicates that the improvement due to complexity is partly attributed to the increased tokens, but increasing the complexity of samples is equivalent to the diversity achieved by four times the token count of simple samples. Moreover, with an equal number of instruction tokens, the evolution of Tree-Instruct yields a 2% higher win rate than those from three iterations of in-depth evolution of WizardLM, demonstrating the superior efficacy of Tree-Instruct in enhancing complexity.

(3) ***Curriculum instruction tuning may not be effective; increasing complexity is all you need***: We implement curriculum learning by progressively training the LLM on increasing levels of difficulty. Initially, we train it on easy-level data with an addition of three nodes, followed by medium-level data with six nodes, and ultimately on hard data with ten nodes added. We observe that, when given the same number of training steps, curriculum learning does outperform training with a mixed difficulty of samples but still falls short compared to training solely on the added ten-node instruction data. This indicates that, as sample complexity increases, the significance of simpler samples diminishes significantly, suggesting that repeating training with complex samples may be sufficient.

2. Related Work

Large Language Models (LLMs), trained on extensive textual datasets, have risen as premier solutions for various NLP tasks (Zhao et al., 2023a). Despite their remarkable performance, these models

[1] https://github.com/tatsu-lab/alpaca_eval

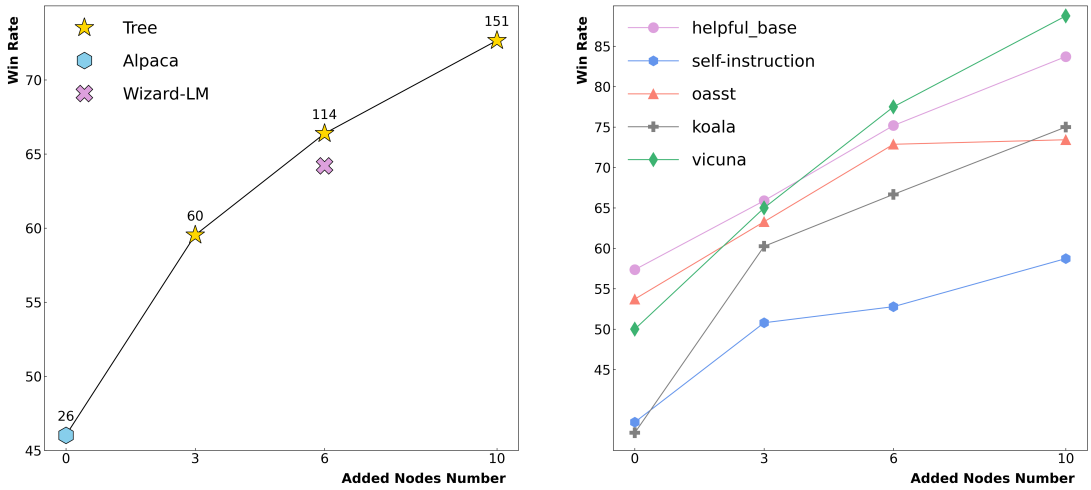


Figure 1: The scaling law of instruction complexity. We experiment with enhancing the complexity of semantic trees for 1,000 Alpaca instructions by adding extra 3, 6, and 10 nodes. We then evaluate models fine-tuned on instruction data of varying complexities against `text-davinci003` in terms of win rate on AlpacaEval (Left). Additionally, we examine win rates on different subsets of AlpacaEval (Right). In the left figure, we indicate above the stars the average token count for instructions of different complexity levels. We also use WizardLM's in-depth deepening as the baseline.

are not without their limitations. These limitations encompass potential misunderstandings of human instructions, the propensity to generate biased content, and the sporadic generation of hallucinated information. Consequently, bringing LLMs in line with human expectations has become a central focal point within the research community (Bai et al., 2022; Song et al., 2023).

To attain this alignment, researchers need to amass high-quality instructional data that authentically mirrors human needs and expectations. A rational starting point for data collection involves the adaptation of existing NLP benchmarks into natural language instructions, like T0 (Sanh et al., 2021), PromptSource (Bach et al., 2022), Super-NaturalInstruction (Wang et al., 2022b), Unnatural Instructions (Honovich et al., 2022) and FLAN (Wei et al., 2021; Longpre et al., 2023) are spearheading this strategy. These benchmarks encompass a wide range of NLP tasks, spanning dialogue, reasoning, and coding, all unified under the realm of language instructions. TÜLU(Wang et al., 2023e) showcases that instructions from NLP tasks significantly bolster the reasoning prowess of aligned LLMs, where the diversity of tasks plays a pivotal role in shaping the capabilities of LLMs.

Nevertheless, a notable trend in NLP datasets is their propensity to emphasize particular skills, consequently yielding instructions that possess a somewhat confined scope. This constraint has the potential to impede their capacity to meet the intricate requirements of real-world applications. In order to tackle these challenges, one possible approach is to formulate instructions via purposeful human annotations. An exemplary precursor to such a corpus is OpenAssistant (Köpf et al., 2023), which comprises over 10k dialogues involving the participation of 13k annotators from around the world. Another remarkable venture into harnessing human-generated instructions through crowdsourcing is ShareGPT [2]. This platform encourages users to contribute and exchange their engaging conversations with ChatGPT and GPT4.

While human annotation ensures both quality and diversity, it becomes challenging to ensure the quantity and complexity of instructional data due to the highly expensive annotation process (Chen et al., 2023c), and the distribution of difficulty levels in human-created instructions tends to skew towards being either easy (Luo et al., 2023). To address this issue, Self-Instruct (Wang et al., 2022a) leverages ChatGPT's in-context learning capability to generate a large volume of instructions from a predefined set of human-annotated instructions spanning diverse topics and task types. Building upon this foundation, LIMA (Zhou et al., 2023) and Alpagasus (Chen et al., 2023a) separately validate the significant impact of data diversity and quality on instructional effectiveness. The selection of thousands of high-quality and diverse instructional examples proves more advantageous in achieving better results compared to using the entire dataset. Further increasing the number of instructions could potentially induce a semantic shift in the LLMs (Al-Shikh et al., 2023). Up to this point, three key metrics within the instructional data—diversity, quality, and quantity—have been elucidated for their im-

[2] 2https://sharegpt.com/

pact on tuning, though exploration into complexity remains insufficient. While WizardLM (Xu et al., 2023) demonstrates that evolving both the complexity and diversity of instructions can lead to performance enhancement, it does not deeply investigate the individual importance of complexity. This paper introduces a method, Tree-Instruct, which enhances instructional complexity while simultaneously constraining thematic consistency to mitigate variations in diversity. Our experiments preliminarily establish a scaling law regarding complexity, show that the improvement resulting from increased complexity isn't solely due to the introduction of more training tokens and illustrate that LLMs only require complex samples for instruction tuning, rather than simple samples serving as foundational padding for curriculum learning.

3. Tree-Instruct

Enhancing the complexity of natural language text seems like a straightforward task for proficient LLMs. For instance, WizardLM utilizes a simple text prompt to complexify instructions as mentioned in Sec. 1. However, due to the extensive pre-training of LLMs on massive corpora, where models predict the next token based on the preceding context, we've noticed that LLMs can often exploit the given instruction by simply continuing the text beyond the initial prompt to artificially amplify complexity. While adding continuation constraints can enhance the complexity of instructions, it simultaneously leads them away from the core thematic focus. This divergence expands the topic and domain, fostering diversity that hinders our ability to solely assess the impact of increased instruction complexity. We leverage GPT-4 to automatically score the consistency (range in $0 \sim 1$) of the instructions before and after implementing in-depth deepening following WizardLM. We found that it only gets a 0.56 alignment score. Furthermore, upon iteratively enhancing the instruction's complexity, the guidance might become ineffective, losing its original essence. For instance, it might cease to present a question, rendering it arduous for the LLM to generate a suitable response. This phenomenon matches with observations made by WizardLM, which prompts them to introduce the Elimination Evolving procedure.

To address this issue, we first consider *what determines the complexity of natural language text*. In linguistics and education, there is a lack of precise scientific consensus on determining the complexity of the text. No single source can precisely summarize a text's complexity. Currently, a widely accepted perspective suggests that qualitative measures of text complexity require an informed judgment of text difficulty based on various factors. The standards use factors like purpose, levels of meaning, structure, language conventions, clarity, and knowledge demands to measure text difficulty [3]. Among these, text structure is a more measurable indicator, as we can convert text sequences into tree structures using mature dependency or semantic tree parsers (Solovyev et al., 2019). Tree structures, prevalent in natural language representations, offer structural insights reflecting human text comprehension (Hancke et al., 2012). Furthermore, we can gauge text complexity accurately by measuring the width and depth of trees, as a deeper and wider grammar tree signifies more intricate sentence structures (Chevalier et al., 2007; Wang et al., 2013).

Inspired by the concept of tree complexity, we propose Tree-Instruct, wherein LLMs directly add a specific number of nodes to the semantic tree of an instruction. This increases the tree's width and depth, thereby enhancing text structure complexity. In detail, Tree-Instruct encompasses three steps:

Step 1: Tree Construction involves semantic parsing, where a structured representation is created from a natural language sentence. This process yields a tree structure for an instruction. For instance, given the instruction "*Implementing effective strategies to curb environmental pollutants in the atmosphere*", we derive an original tree structure Tree-1 as shown in the first tree of Fig. 2.

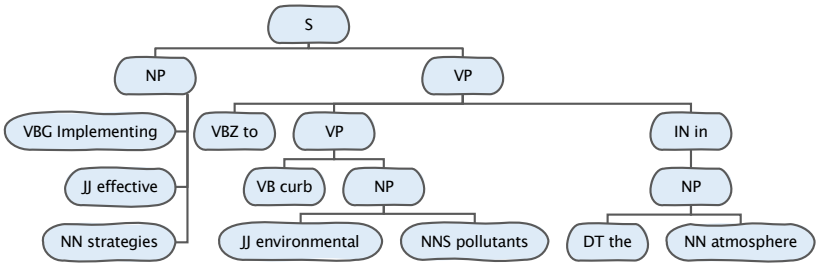
Step 2: Nodes Expansion operates on the acquired tree structure, expanding it in depth or width by adding new nodes, thus influencing the new tree's complexity. We only add meaningful nodes representing nouns or verbs, since words like adjectives or prepositions contribute little to tree complexity. The second tree in Fig. 2 illustrates Tree-2 after adding ten nodes.

Step 3: Tree Sentenceization aims to make LLMs revert the complex new tree structure (Tree-2) back to fluent natural language instruction by introducing connected words.

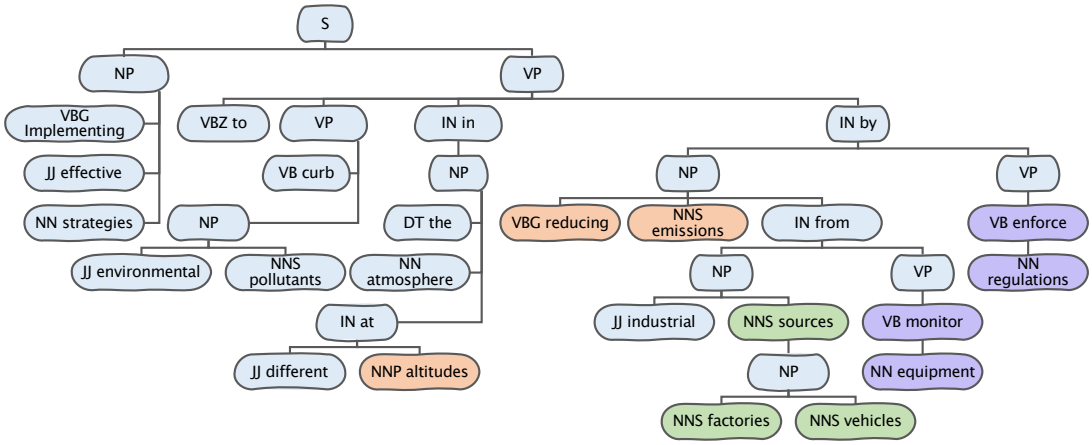
Additionally, we present all three steps into a single prompt, guiding LLMs to implement our requirements step by step without external semantic parsing tools (see Block 3, where "`your_added_number`" indicates the desired number of nodes we aim to add to the tree.) Especially we directly control the complexity by adjusting "`your_added_number`". Visually, with more nodes added, the tree and the instruction become more complex. This gradual increase results in a tree with 3, 6, or 10 additional nodes, progressively increasing the complexity of instructions, as shown in Fig. 2. We also observe that adding nodes to the semantic tree constructs a framework for the original instruction. This approach prevents significant

[3] https://www.generationready.com/wp-content/uploads/2021/04/Beginners-Guide-to-Text-Complexity.pdf

Initial instruction:
Implementing effective strategies to curb environmental pollutants in the atmosphere.



Tree-10-nodes instruction:
Implement effective strategies to curb environmental pollutants in the atmosphere at different altitudes by reducing emissions from industrial sources like factories and vehicles. Additionally, monitor these emissions using specialized equipment and stringently enforce regulations to ensure industries adhere to best practices and environmental standards.



WizardLM Deepening Evolve-iteration-3:
Investigating and formulating intricate methodologies, deeply anchored in cutting-edge quantum and classical scientific principles, to systematically and holistically reduce, monitor, and assess both primary and secondary atmospheric environmental pollutants. This approach is crucial for ensuring sustainable socio-economic progress while actively safeguarding and nurturing our planet's delicate ecological balance.

Figure 2: The instruction generated by different evolving methods: Tree-instruction after adding ten nodes and WizardLM by iteratively *deepening* three times. We also demonstrate how Tree-Instruct enhances the complexity of the original instruction's semantic tree by introducing three nodes (orange), six nodes (green), and ten nodes (purple).

deviations from the main topic. GPT-4's automatic assessment shows that our prompt modifications maintain thematic consistency with a score 0.69.

4. Experiments

In this experiment, our primary objective is to address four key research questions: (1) Can Tree-Instruct, compared to WizardLM's in-depth evolving, better maintain thematic consistency while augmenting complexity? (2) Does increasing the complexity of instructions through Tree-Instruct result in a greater unleashing of LLM's latent potential, i.e., will more intricate instructions yield better outcomes? (3) Given the same token constraints, which approach is better suited for instruction tuning: employing complex yet limited instruction data or opting for simpler but more diverse instructions? (4) Can curriculum-based instruction tuning methods (from simpler to more complex instruction data) yield improvements similar to the substantial enhancements observed in many previous NLP tasks?

Our primary experiments are conducted on Alpaca GPT-4 dataset (Peng et al., 2023), which contains a dataset of 52,000 instruction-following examples responded to by GPT-4 using prompts in Alpaca (Taori et al., 2023). Following LIMA (Zhou et al., 2023), we randomly select 1,000 instruction samples to form Alpaca-1K, serving as the starting point for our evolutionary process. We query `gpt-4` (OpenAI, 2023) to execute Tree-Instruct pro-

Prompt for Tree-Instruct

You are an instruction rewriter. You need to rewrite a given user instruction following Procedures step by step. You MUST ONLY return the NEW instruction you rewrite.

Procedure:
step-1: Parse the old "instruction" to a TREE-1 through Semantic Parsing in the natural language processing field.
step-2: EXPAND the above NEW TREE-1 from depth or width by adding "*your_added_number*" meaningful NEW Nodes as nouns or verbs to form a NEW TREE-2. The new nodes should be constructed with detailed and pertinent information.
step-3: Generate a totally NEW "instruction" based on the expanded NEW TREE-2.

Old instruction: "*your_instruction*"

New instruction:

cess, thereby increasing the complexity of each instruction within Alpaca-1K. In order to analyze the scaling law, we introduce three levels of complexity by augmenting the instructions by adding 3, 6, and 10 additional nodes to the semantic tree of original instructions, respectively. This allows us to observe the impact of these varying complexities on the outcomes. For the modified instructions, we employed `gpt-4` once again to generate corresponding responses. To validate our findings, we replicate the results by applying the in-depth evolving with deepening prompt provided by WizardLM to the same Alpaca-1K instructions.

To demonstrate the scalability of our discoveries to larger datasets, we also conduct experiments on the extensive OpenChat dataset, which comprises 6,206 conversations between humans and GPT4, filtered from ShareGPT (Wang et al., 2023a). We employ the pre-trained LLaMA2 (Touvron et al., 2023) model as the initialization, fine-tuning it on instruction-tuning datasets generated through different methods. Each GPU processes batches of size 2 (for OpenChat evolved data, the batch size is set to 8), and the maximum sequence length was set to 4096. For optimization, we adopt the AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of 1e-4 and a weight decay of 0.1, following the practices established by OpenChat. Using Deepspeed ZeRO-2 stage and 8 A100 GPUs, we train LLaMA2 for 10 epochs on Alpaca-1K and 5 epochs on Openchat-6K data. During inference, a temperature of 0.7 and a top-p value of 0.9 are employed to evaluate all the methods under comparison.

Evaluation Benchmarks We mainly evaluate competing methods on two benchmarks: AlpacaEval and MT-Bench (Zheng et al., 2023). AlpacaEval is an LLM-based automatic evaluation, comprising 805 diverse samples, each showcasing various abilities. In AlpacaEval, responses from different LLM methods are then compared to those from `text-davinci003` by GPT-4 auto-annotator. MT-Bench consists of 80 high-quality multi-turn questions to test multi-turn conversation and instruction-following ability, covering 8 common categories.

4.1. Tree-Instruct is Better for Instruction Complexity Evolution

We start by investigating whether operating on a tree, as opposed to a sequence, better aligns with the intended objectives of the original instruction. Recent studies have introduced the LLMs-as-evaluator paradigm, leveraging LLMs to assess candidate samples, which closely approximates human evaluative agreement (Chen et al., 2023b; Fu et al., 2023; Ji et al., 2023; Zhang et al., 2023). Consequently, we employ `gpt-4` to gauge which approach exhibits greater consistency with the initial instructions. As depicted in Figure 3, the result indicates that employing Tree-Instruct, which entails adding instructions with 6 additional nodes, achieves a higher degree of alignment with the original instructions in 63% of cases, compared to WizardLM's in-depth deepening that undergoes modifications and generates instructions with similar token quantities to Tree-6-nodes. This observation serves as evidence that the presence of a tree structure constraint enables LLMs to more effectively modify instructions within the framework of the original guidance, rather than diverging and incorporating unrelated content. A case study in Fig. 2 also indicates that WizardLM might produce phrases deviated from the original instruction during iterative evolution.

Figure 3: `GPT-4`'s comparison for the consistency preservation between Tree-6-Nodes and WizardLM's in-depth deepening with respect to the original instructions.

Furthermore, our findings demonstrate that Tree-Instruct is more effective than in-depth evolving in eliciting the capabilities of LLMs. We conduct evaluations on the AlpacaEval set for both methods. The evaluations are performed with `gpt-4` as the eval-

Method	helpful-base	self-instruction	oasst	koala	vicuna	Alpaca-Eval	MT-Bench
Alpaca-1K	57.36	38.49	53.72	37.18	50.00	46.02	4.74
Tree-3-nodes	65.89	50.79	63.29	60.25	65.00	59.53 (+13.51)	6.10 (+1.36)
Tree-6-nodes	75.19	52.78	72.87	66.67	77.50	66.38 (+20.34)	6.27 (+1.53)
Tree-10-nodes	83.72	58.73	73.43	75.00	88.75	72.66 (+26.64)	6.43 (+1.68)

Table 1: Analysis of the relationship between instruction complexity and LLM's ability. The numbers represent the win rates vs. `text-davinci003` on various subsets of AlpacaEval and MT-Bench scores.

uator, comparing the win rates of models against `text-davinci003`. As depicted in Table 2, under similar total token counts, Tree-Instruct exhibits a win rate improvement of 2.2 points over WizardLM's in-depth deepening. We attribute this enhancement to Tree-Instruct's adeptness at closely tailoring instructions to the central topic, thereby introducing complexity without deviation.

In contrast, in-depth evolving might deviate from the original theme and introduce irrelevant content, resulting in instructions of inadequate difficulty. Such instructions could potentially hinder LLMs from generating appropriate responses, rendering them less effective in the generation process.

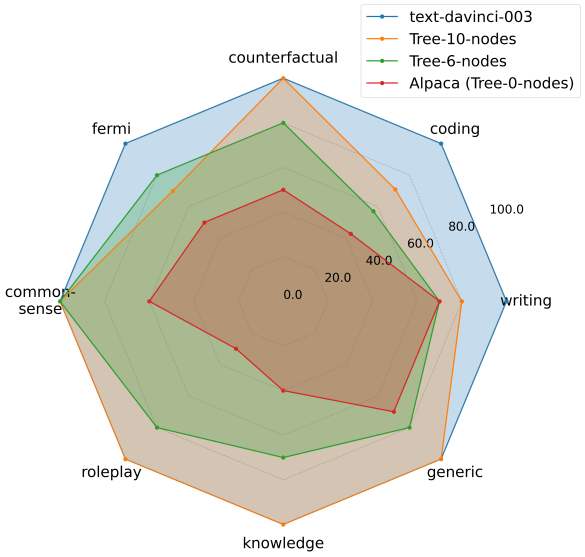


Figure 4: Evaluation of models trained on Alpaca-1K and evolved by adding various nodes vs. `text-davinci003` on categories of the Vicuna test set.

4.2. More Complexity, Better Capability

After demonstrating the effectiveness of Tree-Instruct in enhancing sample complexity, we present a scaling law pertaining to complexity, as depicted in Fig. 1 and Table 1. As the number of nodes gradually increases from Tree-3-Nodes to Tree-6-Nodes and further to Tree-10-Nodes, the model's win rate on the AlpacaEval and scores on MT-Bench benchmarks exhibit a remarkable

Method	Win Rate (%)	Total Token Size
Alpaca-1K	46.02	186,464
Alpaca-4K	55.85	757,042
WizardLM	64.20	556,981
Tree-3-Nodes	59.53	385,760
Tree-6-Nodes	66.38	546,731
Tree-10-Nodes	72.66	608,556

Table 2: Analysis of the scaling laws for complexity and the relationship between win rate and token count with experiments based on LLaMA2.

Method	Win Rate (%)	Token Length
GPT4	95.28	1365
LLaMA2-Chat-70B	92.66	1790
Claude-2	91.36	1069
OpenChat-V3.1	89.49	1484
ChatGPT	89.37	827
Vicuna-33B	88.99	1479
OpenChat-LLaMA2	84.56	1730
OpenChat-LLaMA1	80.87	1632
UltraLM-13B	80.64	1087
WizardLM-13B	75.31	985
Tree-Instruct-LLaMA1	81.82 (+0.95)	1549
Tree-Instruct-LLaMA2	86.19 (+1.63)	1675

Table 3: Win rates of different methods vs. `text-davinci003` on the AlpacaEval leaderboard.

upward trend. This scaling law underscores the significance of complexity within instruction data.

Additionally, we carry out a meticulous evaluation for each skill/category within the Vicuna test sets. These sets are divided into distinct skill sets/categories, allowing for an intricate analysis of the proficiency attained through instruction tuning. Notably, Tree-10-Nodes outperforms Tree-6-Nodes across a majority of categories, encompassing Counterfactual, Roleplay, Knowledge, Generic, and more. Similar trends are evident when comparing Tree-6-Nodes with the original instructions, indicating that augmenting the complexity of Instruction data leads to a comprehensive enhancement in the capabilities of the LLM.

Finally, given that our experimentation is based on 1,000 instances, we extend our investigation to

Method	helpful-base	self-instruction	oasst	koala	vicuna	Overall
Mix-difficulty-training	73.64	50.79	68.62	60.26	70.00	62.59
Hard-to-Easy Curriculum	71.31	56.75	66.49	67.95	71.25	65.05
Easy-to-Hard Curriculum	77.52	57.94	73.93	74.36	85.00	70.95

Table 4: Analysis of mixed difficulty training and curriculum learning. The numbers represent the win rates vs. `text-davinci003` on various subsets of AlpacaEval.

validate the effectiveness of Tree-Instruct across a larger dataset using OpenChat. OpenChat-6K is built upon 6,206 conversations between humans and GPT-4, filtered from around 90K ShareGPT conversations. It has notably achieved top rankings with much less data as an open-source LLM. Since OpenChat involves multi-turn conversations, we specifically complexify instructions through Tree-Instruct by adding three nodes on instructions from single-turn and several last-turn conversations. We ignore instructions that only contain generic and meaningless terms like "stop" or "continue". The Tree-Instruct modification involves 1,147 conversations. In this process, we replace the original data with the evolved version, maintaining the same number of training samples.

As delineated in Table 3, after the complexity evolution of Tree-Instruct, we enhance OpenChat's performance from 80.87% to 81.82% based on LLaMA1, from 84.56% to 86.19% on LLaMA2, underscoring the sustained efficacy of our approach across a larger volume of data. Here the token length refers to the average number of words in outputs generated by models.

4.3. Less but Complex is Better Than More but Simple

While we have demonstrated that increasing the complexity of instruction data can enhance the capabilities of LLMs, a new question arises: Is this improvement merely due to the introduction of more training tokens as complexity increases? Our analysis indicates that the average length of the original Alpaca data, combining both input and output, is 186 tokens. Upon incorporating an additional 10 nodes, this count escalates to 607 tokens – equivalent to a 3.26-fold increase in training data. With this question in mind, we introduce a new baseline: Alpaca-4K, trained with 4,000 samples (additionally sampled 3,000 instances from the original Alpaca data). As shown in Table 2, the total token count[4] of Alpaca-4K surpasses that of Tree-10-Nodes by 24%. Despite this, with the same training steps, a significant 16.8% performance gap in win rate remains. Compared to Alpaca-1K, there is indeed a 9.8% improvement. This suggests that introducing more instruction tokens does enhance model performance. Nonetheless, the effectiveness of diverse yet simple instructions still falls short compared to a smaller quantity of more complex directives.

[4]Total token size refers to the average number of tokenized tokens for input-output pairs

4.4. Curriculum Learning May Be Not Effective for Instruction Tuning

Now, armed with three sets of data featuring increasing difficulty levels and aligned themes, we can delve into an unanswered question in instruction tuning: Is it necessary to train LLM progressively from easy to hard? As depicted in Table 4, we embark on a trial, initially training on Tree-3-Nodes data, followed by Tree-6-Nodes, and finally Tree-10-Nodes. Each segment constitutes one-third of the total training steps. We also devise two baselines: one involving the combined training of all three difficulty levels and another wherein difficult samples are trained prior to the easy ones.

Experimental results reveal that, compared to mixed-difficulty training and training samples from hard to easy, an easy-to-hard curriculum learning approach truly enhances model performance. However, the performance gain from curriculum learning still slightly underperforms exclusively training on Tree-10-Nodes, the hardest dataset we construct. This outcome slightly contrasts with previous observations of curriculum learning. We attribute this variance to the fact that modern LLMs possess parameter counts several times larger than those of earlier models like BERT (Devlin et al., 2019) or T5 (Raffel et al., 2020). With this substantial parameter increase, LLMs are now capable of directly learning from challenging samples, diminishing the need for foundational exposure to simpler samples. The more exposure to challenging samples, the more the model's capabilities are ignited.

5. Conclusion

In this study, we have undertaken a preliminary exploration of the intrinsic relationship between instruction complexity and the ability of large language models to follow human instructions. We propose a controllable method for complicating instructions: Tree-Instruct. We conduct extensive experiments to explore the unknown. The results

reveal the following insights: (1) As the complexity of the instruction data increases, the benefits of instruction tuning continue to amplify. (2) The rise in complexity is partly attributed to additional tokens, yet a few intricate instructions outperform a large number of simpler instructions, all within the same token limit. (3) A curriculum-based instruction tuning, progressing from easier to harder, might not yield the desired effectiveness; embracing increased complexity proves essential. We anticipate that this exploration will supplement existing knowledge regarding the aspects of quality, quantity, diversity, and complexity of instruction data. This contribution aims to assist future researchers in constructing superior instruction data.

6. Limitations

We have conducted extensive experiments to study three unexplored aspects targeting the complexity of instruction-turning data: (1) the scaling law, (2) the impact of additional training tokens, and (3) curriculum learning. Kindly notice that, in Tree-Instruct, responses adapted to original instructions are no longer capable of fulfilling the needs of complex instructions, which means that users need to regenerate responses. Moreover, due to constraints in time and resources, such as computing resources or accessing GPT-4, there are still some unexplored questions: 1. Will instruction evolution ever reach a point of convergence? 2. How can we adapt Tree-Instruct for complex tasks, such as mathematics or coding, that require intricate reasoning? 3. Are larger language models still sensitive to the complexity of instruction-tuning data? We plan to delve into these areas in our subsequent research.

7. Ethical Statement

This paper studies the complexity of instruction-tuning data for large language models and proposes a novel method Tree-Instruct, to control the complexity evolution. All the datasets and models involved in this paper are publicly available. The prompts we design are also harmless and unbiased, leading to healthy and objective training data. There are no direct ethical concerns in our study.

8. Bibliographical References

Waseem AlShikh, Manhal Daaboul, Kirk Goddard, Brock Imel, Kiran Kamble, Parikshith Kulkarni, and Melisa Russak. 2023. Becoming self-instruct: introducing early stopping criteria for minimal instruct tuning. *arXiv preprint arXiv:2307.03692*.

Anthony McEnery and others. 2004. *The EMILLE/CIIL Corpus*. EMILLE (Enabling Minority Language Engineering) Project. distributed via ELRA: ELRA-Id W0037, ISLRN 039-846-040-604-0.

Stephen H Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, et al. 2022. Promptsource: An integrated development environment and repository for natural language prompts. *arXiv preprint arXiv:2202.01279*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2018. Identifying and controlling important neurons in neural machine translation. *arXiv preprint arXiv:1811.01157*.

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. 2020. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Adithya Bhaskar, Alex Fabbri, and Greg Durrett. 2023. Prompted opinion summarization with gpt-3.5. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9282–9300.

BSI. 1973a. *Natural Fibre Twines*, 3rd edition. British Standards Institution, London. BS 2570.

BSI. 1973b. Natural fibre twines. BS 2570, British Standards Institution, London. 3rd. edn.

A. Castor and L. E. Pollux. 1992. The use of user modelling to guide inference and learning. *Applied Intelligence*, 2(1):37–53.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. 2023a. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yi Chen, Rui Wang, Haiyun Jiang, Shuming Shi, and Ruifeng Xu. 2023b. Exploring the use of large language models for reference-free text quality evaluation: A preliminary empirical study.

Zhihong Chen, Feng Jiang, Junying Chen, Tiannan Wang, Fei Yu, Guiming Chen, Hongbo Zhang, Juhao Liang, Chen Zhang, Zhiyi Zhang, et al. 2023c. Phoenix: Democratizing chatgpt across languages. *arXiv preprint arXiv:2304.10453*.

J.L. Chercheur. 1994. *Case-Based Reasoning*, 2nd edition. Morgan Kaufman Publishers, San Mateo, CA.

Fanny Chevalier, David Auber, and Alexandru Telea. 2007. Structural analysis and visualization of c++ code evolution using syntax trees. In *Ninth international workshop on Principles of software evolution: in conjunction with the 6th ESEC/FSE joint meeting*, pages 90–97.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*.

N. Chomsky. 1973. Conditions on transformations. In *A festschrift for Morris Halle*, New York. Holt, Rinehart & Winston.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialog state tracking. *arXiv preprint arXiv:2106.00291*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Umberto Eco. 1990. *The Limits of Interpretation*. Indian University Press.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. 2020. Multi-modal transformer for video retrieval. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 214–229. Springer.

Julia Hancke, Sowmya Vajjala, and Detmar Meurers. 2012. Readability classification for german using lexical, syntactic, and morphological features. In *Proceedings of COLING 2012*, pages 1063–1080.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and understanding the effectiveness of bert. *arXiv preprint arXiv:1908.05620*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

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

Paul Gerhard Hoel. 1971a. *Elementary Statistics*, 3rd edition. Wiley series in probability and mathematical statistics. Wiley, New York, Chichester. ISBN 0 471 40300.

Paul Gerhard Hoel. 1971b. *Elementary Statistics*, 3rd edition, Wiley series in probability and mathematical statistics, pages 19–33. Wiley, New York, Chichester. ISBN 0 471 40300.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, et al. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *arXiv preprint arXiv:2305.08322*.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. 2022. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. *arXiv preprint arXiv:2212.12017*.

Siddhartha Jain, Xiaofei Ma, Anoop Deoras, and Bing Xiang. 2023. Self-consistency for open-ended generations. *arXiv preprint arXiv:2307.06857*.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the benefits of training expert language models over instruction tuning. *arXiv preprint arXiv:2302.03202*.

Otto Jespersen. 1922. *Language: Its Nature, Development, and Origin*. Allen and Unwin.

Yunjie Ji, Yan Gong, Yiping Peng, Chao Ni, Peiyan Sun, Dongyu Pan, Baochang Ma, and Xiangang Li. 2023. Exploring chatgpt's ability to rank content: A preliminary study on consistency with human preferences.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Khalid Choukri and Niklas Paullson. 2004. *The OrienTel Moroccan MCA (Modern Colloquial Arabic) database*. distributed via ELRA: ELRA-Id ELRA-S0183, ISLRN 613-578-868-832-2.

Tom Kocmi and Christian Federmann. 2023. Large language models are state-of-the-art evaluators of translation quality. *arXiv preprint arXiv:2302.14520*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. 2023. Openassistant conversations–democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*.

Hyungtae Lee and Heesung Kwon. 2017. Going deeper with contextual cnn for hyperspectral image classification. *IEEE Transactions on Image Processing*, 26(10):4843–4855.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Minghao Li, Feifan Song, Bowen Yu, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A benchmark for tool-augmented llms. *arXiv preprint arXiv:2304.08244*.

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

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

Song Liu, Haoqi Fan, Shengsheng Qian, Yiru Chen, Wenkui Ding, and Zhongyuan Wang. 2021. Hit: Hierarchical transformer with momentum contrast for video-text retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11915–11925.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*.

Aravindh Mahendran and Andrea Vedaldi. 2015. Understanding deep image representations by

inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196.

Niklas Muennighoff, Alexander M Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling data-constrained language models. *arXiv preprint arXiv:2305.16264*.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

OpenAI. 2022. Introducing chatgpt.

R OpenAI. 2023. Gpt-4 technical report. *arXiv*, pages 2303–08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002a. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002b. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander Hauptmann, Joao Henriques, and Andrea Vedaldi. 2020. Support-set bottlenecks for video-text representation learning. *arXiv preprint arXiv:2010.02824*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of bert in ranking. *arXiv preprint arXiv:1904.07531*.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Roventini, Adriana and Marinelli, Rita and Bertagna, Francesca. 2016. *ItalWordNet v.2*. ILC-CNR for CLARIN-IT repository hosted at Institute for Computational Linguistics "A. Zampolli", National Research Council, in Pisa, ISLRN 532-206-426-067-2. PID http://hdl.handle.net/20.500.11752/ILC-62. Note: You don't really need both an ISLRN and another PID, but it can't hurt.

Mrinmaya Sachan and Eric Xing. 2016. Easy questions first? a case study on curriculum learning for question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 453–463.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*.

Vighnesh Shiv and Chris Quirk. 2019. Novel positional encodings to enable tree-based transformers. *Advances in neural information processing systems*, 32.

Charles Joseph Singer, E. J. Holmyard, and A. R. Hall, editors. 1954–58. *A history of technology*. Oxford University Press, London. 5 vol.

Valery Solovyev, Marina Solnyshkina, Vladimir Ivanov, and Ivan Rygaev. 2019. Computing syntactic parameters for automated text complexity assessment. In *CEUR Workshop Proceedings*, volume 2475, pages 62–71.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2023. Preference ranking optimization for human alignment. *arXiv preprint arXiv:2306.17492*.

Alessandro Sordoni, Xingdi Yuan, Marc-Alexandre Côté, Matheus Pereira, Adam Trischler, Ziang Xiao, Arian Hosseini, Friederike Niedtner, and Nicolas Le Roux. 2023. Deep language networks: Joint prompt training of stacked llms using variational inference. *arXiv preprint arXiv:2306.12509*.

Speecon Consortium. 2011. *Catalan Speecon database*. SpeeCon. Speecon Project, distributed via ELRA: ELRA-Id S0327, Speecon resources, 1.0, ISLRN 935-211-147-357-5.

Jannik Strötgen and Michael Gertz. 2012. Temporal tagging on different domains: Challenges, strategies, and gold standards. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3746–3753, Istanbul, Turkey. European Language Resource Association (ELRA).

S. Superman, B. Batman, C. Catwoman, and S. Spiderman. 2000. *Superheroes experiences with books*, 20th edition. The Phantom Editors Associates, Gotham City.

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.

Yi Tay, Jason Wei, Hyung Won Chung, Vinh Q Tran, David R So, Siamak Shakeri, Xavier Garcia, Huaixiu Steven Zheng, Jinfeng Rao, Aakanksha Chowdhery, et al. 2022. Transcending scaling laws with 0.1% extra compute. *arXiv preprint arXiv:2210.11399*.

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

Guan Wang, Sijie Cheng, Qiying Yu, and Changling Liu. 2023a. OpenChat: Advancing Open-source Language Models with Imperfect Data. *10.5281/zenodo.8105775*.

Jiaan Wang, Yunlong Liang, Fandong Meng, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023b. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023c. Large language models are not fair evaluators.

Xiangli Wang, Yi Zhang, Yusuke Miyao, Takuya Matsuzaki, and Jun'ichi Tsujii. 2013. Deep context-free grammar for chinese with broad-coverage. In *Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing*, pages 11–19.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2023d. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023e. How far can camels go? exploring the state of instruction tuning on open resources. *arXiv preprint arXiv:2306.04751*.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023f. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023g. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Xiangpeng Wei, Haoran Wei, Huan Lin, Tianhao Li, Pei Zhang, Xingzhang Ren, Mei Li, Yu Wan, Zhiwei Cao, Binbin Xie, et al. 2023. Polylm: An open source polyglot large language model. *arXiv preprint arXiv:2307.06018*.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27263–27277. Curran Associates, Inc.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xinghua Zhang, Bowen Yu, Haiyang Yu, Yangyu Lv, Tingwen Liu, Fei Huang, Hongbo Xu, and Yongbin Li. 2023. Wider and deeper llm networks are fairer llm evaluators. *arXiv preprint arXiv:2308.01862*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023a. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. 2019. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*.

Yingxiu Zhao, Bowen Yu, Haiyang Yu, Bowen Li, Chao Wang, Fei Huang, Yongbin Li, and Nevin L Zhang. 2023b. Causal document-grounded dialogue pre-training. *arXiv preprint arXiv:2305.10927*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2014. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

Yikai Zhou, Baosong Yang, Derek F Wong, Yu Wan, and Lidia S Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the association for computational linguistics*, pages 6934–6944.

Qingqing Zhu, Xiuying Chen, Pengfei Wu, JunFei Liu, and Dongyan Zhao. 2021. Combining curriculum learning and knowledge distillation for dialogue generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1284–1295.