# Synthetic Data (Almost) from Scratch: Generalized Instruction Tuning for Language Models

Haoran Li*, Qingxiu Dong*, Zhengyang Tang*, Chaojun Wang*, Xingxing Zhang*, Haoyang Huang*
Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng
Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, Furu Wei

https://aka.ms/GeneralAI

## Abstract

We introduce *Generalized Instruction Tuning* (called GLAN), a general and scalable method for instruction tuning of Large Language Models (LLMs). Unlike prior work that relies on seed examples or existing datasets to construct instruction tuning data, GLAN exclusively utilizes a pre-curated taxonomy of human knowledge and capabilities as input and generates large-scale synthetic instruction data across all disciplines. Specifically, inspired by the systematic structure in human education system, we build the taxonomy by decomposing human knowledge and capabilities to various fields, sub-fields and ultimately, distinct disciplines semi-automatically, facilitated by LLMs. Subsequently, we generate a comprehensive list of subjects for every discipline and proceed to design a syllabus tailored to each subject, again utilizing LLMs. With the fine-grained key concepts detailed in every class session of the syllabus, we are able to generate diverse instructions with a broad coverage across the entire spectrum of human knowledge and skills. Extensive experiments on large language models (e.g., Mistral) demonstrate that GLAN excels in multiple dimensions from mathematical reasoning, coding, academic exams, logical reasoning to general instruction following without using task-specific training data of these tasks. In addition, GLAN allows for easy customization and new fields or skills can be added by simply incorporating a new node into our taxonomy.

## 1 Introduction

Large Language Models (LLMs) have enabled unprecedented capabilities to understand and generate text like humans. By scaling up model size and data size [KMH+20, HBM+22], LLMs are better at predicting next tokens and prompting to perform certain tasks with a few demonstrations [BMR+20]. However, these capabilities do not directly translate to better human instruction following [OWJ+22]. Instruction tuning [WBZ+21] bridges this gap through fine-tuning LLMs on instructions paired with human-preferred responses.

Prior work constructs instruction tuning data from seed examples or existing datasets. Initially, natural language processing (NLP) datasets described via instructions are used to fine-tune LLMs and the resulting LLMs can generalize on unseen (NLP) tasks [WBZ+21]. However, there are only thousands of NLP tasks [WMA+22, LHV+23] available, which limits the tuned LLMs to generalize in real-world scenarios [XSZ+23]. Self-instruct [WKM+22] is a cost effective method for creating synthetic instruction tuning datasets, which starts from a small pool of human-written seed instructions and

*Equal contribution. X. Zhang (xingxing.zhang@microsoft.com), H. Huang, S. Huang, X. Huang, Z. Huang, D. Zhang, X. Wang, S. Chen, L. Dong and F. Wei are with Microsoft. H. Li and W. Lu are with Singapore University of Technology and Design. Q. Dong, X. Cheng and Z. Sui are with Peking University. Z. Tang and B. Wang are with Chinese University of Hong Kong, Shenzhen. C. Wang and W. Lam are with Chinese University of Hong Kong. Y. Gu is with Tsinghua University.

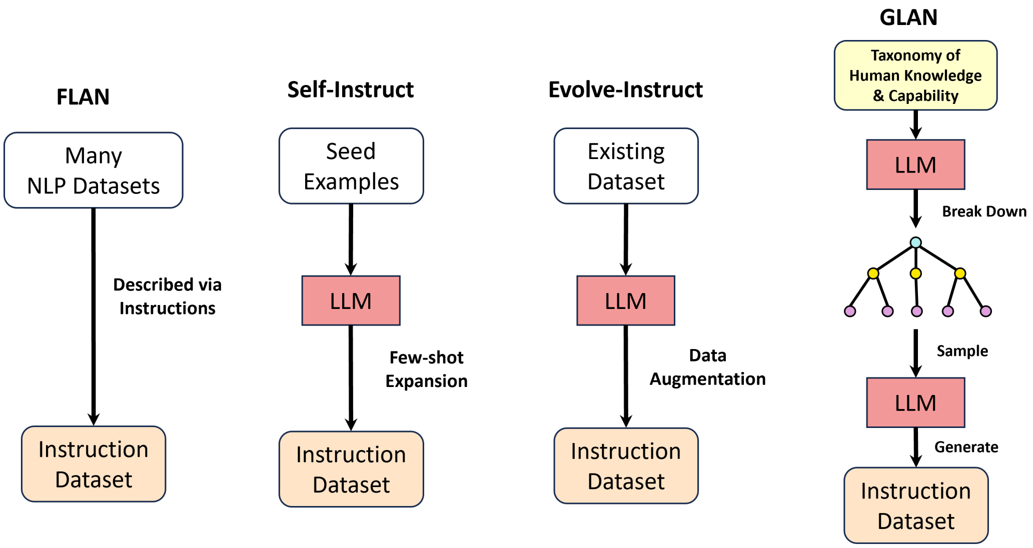

Figure 1: Comparing GLAN with FLAN, Self-Instruct and Evolve-Instruct. The inputs of FLAN, Self-Instrct and Eovlve-Instruct are either seed examples or existing datasets, which limits the scope of domains of instructions that these methods can generate. GLAN takes the taxonomy of human knowledge & capabilities as input to ensure the broad coverage of generated instructions in various domains. This taxonomy is then broken down into smaller pieces and recombined to generate diverse instruction data.

generates new instructions by few-shot prompting an LLM (e.g., `text-davinci-002`) with randomly selected instructions from the pool. Unfortunately, the diversity of generated instructions is still an issue, since few-shot prompting tends to generate new instructions similar to its demonstrations. In addition, the process of creating high-quality seed instructions requires considerable human effort and expertise. Evolve-Instruct [XSZ$^+$23] improves self-instruct by augmenting existing instruction tuning datasets with different rewriting operations using LLMs, which is essentially data argumentation. Consequently, the scope of domains or tasks that these augmented datasets can cover is limited by the original input datasets. See Figure 1 for illustrations of these methods described above. There are also studies concentrated on developing instruction tuning datasets tailored to particular domains or tasks. For instance, [LSX$^+$23] creates datasets targeting mathematical reasoning. In contrast, [Cha23] and [LXZ$^+$23] primarily focus on coding-related tasks. All these methods above cannot produce instruction datasets which are generally applicable to a wide range of domains.

How to create a *general* instruction tuning dataset? We draw inspiration from the systematic structure in human education system. The structure of human education includes several levels, starting from early childhood education up to higher education and beyond [wik23]. Within each level, a student acquires knowledge, skills and values in a systematical process. The courses a student learn from primary school to college covers a broad range of knowledge and skills, which facilitates the development of a diverse array of abilities. We believe the systemic framework of the human education system has the potential to help the generation of high-quality and *general* instruction data, which spans a diverse range of disciplinary areas.

In this paper, we introduce a generalized instruction tuning paradigm GLAN (shorthand for **G**eneralized Instruction-Tuning for **L**arge **LAN**guage Models) to generate synthetic instruction tuning data almost from scratch. Unlike existing work [XSZ$^+$23, LXZ$^+$23, LSX$^+$23, MMJ$^+$23], GLAN exclusively utilizes a pre-curated taxonomy of human knowledge and capabilities as input and generates large-scale instruction data systematically and automatically across all disciplines. Specifically, inspired by the structure in human education system, the input taxonomy is constructed by decomposing human knowledge and capabilities to various fields, sub-fields and ultimately, distinct disciplines semi-automatically, facilitated by LLMs and human verification. The cost of human verification process is low due to the limited number of disciplines in the taxonomy. As shown in Figure 1, we then further beak down these disciplines to even smaller units. We continue to generate a comprehensive list of subjects for every discipline and proceed to design a syllabus tailored to each subject, again utilizing LLMs. With the fine-grained key concepts detailed in every class session of the syllabus, we can first sample from them and then generate diverse instructions with a broad coverage across the entire spectrum of human knowledge and skills. The process described above mirrors the human educational system, where educators in each discipline craft a series of subjects for student learning. Instructors then develop a syllabus for each subject, breaking down the content into specific class sessions. These sessions are then further divided into core concepts that students must comprehend and internalize. Based on these detailed core concepts outlined in the syllabus, teaching materials and exercises are subsequently created, which are our instruction tuning data.

GLAN is general, scalable and customizable. GLAN is a general method, which is task-agnostic and is capable of covering a broad range of domains. GLAN is scalable. Similar to [WKM+22, XSZ+23], GLAN generate instructions using LLMs, which can produce instructions in a massive scale. Moreover, the input of GLAN is a taxonomy, which is generated by prompting an LLM and human verification, requiring minimal human effort. GLAN allows for easy customization. New fields or skills can be added by simply incorporating a new node into our taxonomy. Note that each node of the taxonomy can be expanded independently, which means that we only need to apply our method to the newly added nodes without re-generating the entire dataset. Extensive experiments on large language models (e.g., Mistral) demonstrate that GLAN excels in multiple dimensions from mathematical reasoning, coding, academic exams, logical reasoning to general instruction following without using task-specific training data of these tasks.

## 2 GLAN: Generalized Instruction-Tuned Language Models

GLAN aims to create synthetic instruction data covering various domains of human knowledge and capabilities in large scale. As shown in Algorithm 1, we first build a taxonomy of human knowledge and capabilities using frontier LLMs (i.e., `GPT-4`) and human verification. The taxonomy naturally breaks down human knowledge and capabilities to *fields*, *sub-fields* and ultimately different *disciplines* (see Section 2.1). The following steps are fully autonomous facilitated by `GPT-4` (or `GPT-3.5`). Then for each discipline, we again instruct `GPT-4` to further decompose it to a list of subjects within this discipline (Section 2.2). Similar to an instructor, `GPT-4` continues to design a syllabus for each subject, which inherently breaks a subject to various class sessions with key concepts students need to master (Section 2.3). With obtained class sessions and key concepts, we are ready to construct synthetic instructions. We prompt `GPT-4` to generate homework questions based on randomly sampled class sessions and key concepts as well as the syllabus (Section 2.4). We recursively decompose human knowledge and capabilities to smaller units until atomic-level components (i.e., class sessions and key concepts). We expect by randomly combining these class sessions and key concepts to ensure the coverage and diversity of synthetic instructions.

**Algorithm 1** GLAN Instruction Generation

```
$\mathbb{D} \leftarrow$ build_taxonomy()    ▷ build a taxonomy and return a list of disciplines (Section 2.1)
$\mathbb{L} \leftarrow \varnothing$
for each discipline $d \in \mathbb{D}$ do
    $\mathbb{S} \leftarrow$ generate_subjects($d$)    ▷ Obtain a list of subjects in $d$ (Section 2.2)
    for each subject $s \in \mathbb{S}$ do
        $\mathcal{A} \leftarrow$ generate_syllabus($s, d$)    ▷ Return syllabus $\mathcal{A}$ for $s$ (Section 2.3)
        $\mathbb{C}, \mathbb{K} \leftarrow$ extract_class_details($\mathcal{A}$)    ▷ Extract class sessions and key concepts (Section 2.3)
        $\mathbb{Q} \leftarrow$ generate_instructions($\mathcal{A}, \mathbb{C}, \mathbb{K}, d$)    ▷ Generate instructions by sampling class sessions and key concepts (Section 2.4)
        $\mathbb{L} \leftarrow \mathbb{L} \cup \mathbb{Q}$
    end for
end for
return $\mathbb{L}$
```

### 2.1 Taxonomy of Human Knowledge and Capabilities

We build a taxonomy of human knowledge and capabilities to guide the generation of synthetic instructions. Therefore, its coverage is important. On the other hand, it is also essential to make the taxonomy highly extensible, since the preferred capabilities of LLMs may change over time. In the first step, we propose to generate the taxonomy by prompting `GPT-4` with a set of different instructions (e.g., `list all fields of human knowledge and capabilities`). Then, we do human post-editing to ensure its correctness and completeness. Due to the limited number of fields, sub-fields, and disciplines in our taxonomy, the cost of human verification is reasonably low. Another advantage of human post-editing is that we can easily add new fields or disciplines to the taxonomy as needed.

Our taxonomy currently covers a diverse range of knowledge and capabilities in both academic education and vocational training. The top level of the taxonomy contains *fields* such as *Natural Sciences*, *Humanities* or *Services* (vocational training). These fields branch out to various *sub-fields* and/or *disciplines* such as *Chemistry*, *Sociology* or *Retailing*. We keep breaking down nodes of the taxonomy until *disciplines* and we leave the breaking down of disciplines to automatic methods described in following sections. By collecting the leaf nodes of the taxonomy, we obtain a list of disciplines $\mathbb{D} = \{d_1, d_2, \ldots, d_M\}$.

### 2.2 Subject Generator

As in Algorithm 1, for each discipline $d$, we aim to extract the list of subjects in it through prompt engineering. Specifically, we instruct `GPT-4` to `act as an education expert of discipline` $d$ `and design a list of subjects a student should learn`. The completion of `GPT-4` contains a comprehensive list of subjects and their meta data (e.g., level, introduction and subtopics of the subject) in unstructured text format, which can not be directly used in subsequent steps. We therefore used another round of prompting to convert the completion to `jsonl` format.

````
Awesome! Transform the above to jsonl format so that it is easier for a computer to understand. Put
    ↪ the jsonl output between "```" "```" tags

For each line, use the keys "subject_name", "level" and "subtopics"
````

It is worth noting that generating a subject list in `jsonl` format using a single prompt is feasible. However, we refrain to do so, because we observe that incorporating additional formatting instructions directly into the prompt can compromise the quality of the resulting subject list. These extracted subjects (as well as their meta data) $\mathbb{S} = \{s_1, s_2, \ldots, s_N\}$ can be subsequently used in next steps. For each $s \in \mathbb{S}$, let `s.name`, `s.level` and `s.subtopics` denote the name, grade level and subtopics of subject $s$, respectively. We can apply the above prompts multiple times to ensure better coverage of subjects within this discipline.

### 2.3 Syllabus Generator

For each subject $s$, we have already extracted its name (`s.name`), grade level (`s.level`) and a small set of included sub-topics (`s.subtopics`) in a structured format. In this section, we aim to further segment each subject into smaller units, making them more suitable for creating homework assignments. We consult `GPT-4` to design a syllabus for this subject. We opt for syllabus generation for the following reasons:

- A syllabus essentially breaks down the main topic of a subject into smaller segments in a hierarchical manner. Specifically, each subject comprises several class sessions, and each session covers a variety of sub-topics and key concepts.
- A syllabus provides an introduction, objectives, and expected outcomes of a subject, which are inherently useful for formulating homework questions.

We instruct `GPT-4` to 1) design a syllabus based on its meta data (`s.level`, `s.name` and `s.subtopics`); 2) break the subject to different class sessions; 3) provide details for each class session with a description and detailed key concepts students need to master. Let $\mathcal{A}$ denote the generated syllabus.

The resulting syllabus $\mathcal{A}$ is in unstructured text format. However, class sessions names and key concepts of each class are required in the instruction generation step (see Algorithm 1). Similar to the process of subject list extraction in Section 2.2, we again extract these meta data of each class session by prompting GPT-4. As a result, we obtain a list of class sessions $\mathbb{C} = \{c_1, c_2, \ldots, c_{|\mathbb{C}|}\}$ and their corresponding key concepts $\mathbb{K} = \{\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_{|\mathbb{C}|}\}$.

### 2.4 Instruction Generator

Given a syllabus $\mathcal{A}$ as well as a list of its class sessions $\mathbb{C}$ and their associated key concepts $\mathbb{K}$, we are ready to generate homework questions and their answers. To generate diverse homework questions, we first sample one or two class session names from $\mathbb{C}$ and one to five key concepts under these selected class sessions. Let $\hat{\mathbb{C}}$ denote the selected class session names and $\hat{\mathbb{K}}$ the selected key concepts. Then we prompt GPT-4 (or GPT-3.5) to generate a homework question given the selected class sessions $\hat{\mathbb{C}}$ and key concepts $\hat{\mathbb{K}}$ as well as the syllabus $\mathcal{A}$. We intend to give GPT-4/3.5 more context (e.g., what students have already learned in previous sessions) when creating assignments. Therefore, we additionally instruct GPT to consider that student have learned up to class sessions $\hat{\mathbb{C}}$ when crafting homework and try to leverage multiple key concepts across different class sessions.

**Sampling Class Sessions and Key Concepts** In a single syllabus, there are numerous class sessions and key concepts. We have two strategies to sample from them. In the first strategy, we generate assignments from a single class session. Therefore, we have only one class session name. Suppose we have $m$ key concepts in total in this session. We randomly sample one to five key concepts from the $m$ key concepts, which mean we have totally $\sum_{i=1}^{5}\binom{m}{i}$ combinations. In this strategy, we focus on creating *basic* homework questions. To make the resulting questions more challenging (combine knowledge from multiple class sessions), we propose a second strategy to combine key concepts from two class sessions in the second strategy. We intend to generate questions leverage knowledge from two different class sessions. Suppose we have $m_1$ and $m_2$ key concepts in the first and second class sessions, respectively. We can have $\sum_{i=2}^{5}\binom{m_1+m_2}{i} - \sum_{i=2}^{5}\binom{m_1}{i} - \sum_{i=2}^{5}\binom{m_2}{i}$ different combinations, which is significantly more than that of the first strategy. We use both strategies to ensure our created questions are diverse in difficulty levels.

**Answer Generation** After we generate questions in previous steps, we simply send these questions to GPT-3.5 and collect answers. We use GPT-3.5 for answer generation, because we find the quality of generated answers from GPT-3.5 is sufficient and using GPT-3.5 is significantly faster than GPT-4. The resulting question-answer pairs are our instruction tuning data. With huge amount of question-answer pairs ranging from different disciplines with various difficulty levels, we expect the resulting LLM can excel in a wide range of tasks.

## 3 Experiments

### 3.1 Data Generation

**Taxonomy Creation** By asking GPT-4 to create a taxonomy of human knowledge and capabilities, we end up with a set of fields, sub-fields and disciplines that cover a broad range of domains in human knowledge and capabilities. Next, we ask human annotators to decide whether these elements in the taxonomy should be kept or not in order to reduce the redundancy of the taxonomy while maintaining its correctness. Note that if a field or sub-field is marked as *remove*, we remove its descendant as well. We kept 126 *disciplines* after majority voting. Note that it is feasible to manually add extra disciplines, sub-fields or fields whenever necessary.

**Subject and Syllabus Generation** During the subject list and syllabus generation, we prompt GPT-4 and employ nucleus sampling [HBD+19] with temperature $T = 1.0$ and top-$p = 0.95$ to encourage diversity. We do not use GPT-3.5-turbo since some subjects belong to the long-tail distribution which may not be effectively modeled by GPT-3.5-turbo. To ensure diversity and completeness of the generated subjects, We query GPT-4 10 times for each discipline (Section 2.2). There are 100 to 200 subjects for each discipline on average. It is worth noting that the same subjects may appear in different disciplines. For instance, the subject *calculus* is both in physics and

| Model | $\lvert\theta\rvert$ | HumanE | MBPP | GSM8K | MATH | BBH | ARC-E | ARC-C | MMLU |
|---|---|---|---|---|---|---|---|---|---|
| GPT-4 | – | 88.4 | 80.0 | 92.0 | 52.9 | 86.7 | 95.4 | 93.6 | 86.4 |
| GPT-3.5-turbo | – | 72.6 | 70.8 | 74.1 | 37.8 | 70.1 | 88.9 | 83.7 | 70.0 |
| LLaMA2 | 7B | 12.8 | 36.2 | 15.4 | 4.2 | 39.6 | 74.6 | 46.3 | 45.9 |
| Orca 2 | 7B | 17.1 | 28.4 | 55.7 | 10.1 | 42.8 | 87.8 | 78.4 | 53.9 |
| WizardLM v1.2 | 13B | 31.7 | 47.9 | 46.8 | 9.0 | 48.4 | 74.2 | 50.2 | 52.7 |
| Mistral | 7B | 28.0 | 50.2 | 43.4 | 10.0 | 56.1 | 79.5 | 53.9 | 62.3 |
| Mistral Instruct | 7B | 46.7 | 31.7 | 24.4 | 8.2 | 46.0 | 76.9 | 52.0 | 53.7 |
| MetaMath Mistral | 7B | 35.4 | 48.6 | 77.7 | 28.2 | 55.7 | 77.3 | 51.0 | 61.0 |
| WizardMath v1.1 | 7B | **51.2** | 54.1 | **83.2** | **33.0** | 58.2 | 79.8 | 53.2 | 60.3 |
| Mistral CodeAlpaca | 7B | 35.4 | 50.2 | 34.6 | 8.3 | 56.1 | 79.1 | 54.2 | 60.9 |
| GLAN | 7B | 48.8 | **57.6** | 80.8 | 32.7 | **60.7** | **90.7** | **81.1** | **62.9** |

Table 1: Main results on Mathematical Reasoning, Coding, Logical Reasoning and Academic Exam benchmarks. Best results are in boldface, while second best results are underscored.

mathematics. We do not de-duplicate those subjects, since it may reflects their importance in human knowledge.

Given a subject in a specified discipline, we query `GPT-4` for only one time to design a syllabus (see details in section 2.3). The temperature and top-$p$ are still set to 1.0 and 0.95, respectively. The number of class sessions contained in each syllabus varies from 10 to 30 and each class session contains around five key concepts.

**Instruction Generation** Each instruction data consists of a question and its answer. We choose to generate questions and answers separately since we observed that separate generations lead to better quality. After question generation with `GPT-4`, each question is then answered by `GPT-3.5-turbo` with temperature $T = 0.7$, top-$p = 0.95$ (we use a lower temperature in order to make the resulting answers more accurate). We use `GPT-3.5-turbo` instead of `GPT-4` for answer generation, because `GPT-3.5-turbo` is significantly faster with reasonably good results. We generate 10 million instruction-response pairs in total and then we do training data decontamination. Specifically, the training instruction-response pairs are decontaminated by removing pairs that contain questions or input prompts from the test and training (if any) sets of benchmarks we evaluate. We exclude training set of benchmarks we evaluate to verify the generatlization capability of our synthetic data.

### 3.2 Model Training

We employ Mistral 7B [JSM+23] as our base model. During training, we concatenate each instruction and response pair to a single sequence and only compute loss on the response tokens. We train our model for three epochs with a learning rate of $3e - 6$. The batch size is set to 512 instruction-response pairs. We use a cosine learning rate schedule and we start with a linear warm-up of 1000 steps and the final learning rate is reduced to 0.

### 3.3 Benchmark Evaluation

The instruction data GLAN generated spans a wide range of subjects. We evaluate its effectiveness in mathematical reasoning, coding, logical reasoning and academic exams.

**Mathematical Reasoning** Mathematics is a common subject in many different disciplines. Hence, it is necessary to test the math reasoning ability of GLAN. We choose the two popular benchmarks for evaluation (i.e., GSM8K [CKB+21] and MATH [HBK+21]). Grade School Math Word Problems (GSM8K [CKB+21]) is a high quality math problem dataset that measures the basic multi-step mathematical reasoning ability. It contains around 7k problems for training and 1K test problems for evaluation. Mathematics Aptitude Test of Heuristics dataset (MATH [HBK+21]) is a challenging math dataset that contains mathematics competition problems from AMC 10, AMC 12, AIME and so on. The 7.5k training and 5K test problems cover seven math subjects, i.e., Prealgebra, Precalculus,

| Model | ARC-E | ARC-C | MMLU |  |  |  |
|---|---|---|---|---|---|---|
|  |  |  | STEM | Humanities | Social Sciences | Other |
| Mistral | 79.5 | 53.9 | 52.0 | 56.5 | 73.3 | 70.1 |
| GLAN | **90.7** | **81.1** | **60.1** | 54.9 | 71.8 | 68.6 |

Table 2: Detailed Results on Academic Exam benchmarks.

Algebra, Intermediate Algebra, Number Theory, Counting and Probability and Geometry. Note that GLAN does not use any examples in the training set of GSM8K or MATH. Following [LSX+23], we report 0-shot setting results for GLAN.

**Coding** To evaluate the coding capability of GLAN, we opt for two coding benchmarks HumanEval [CTJ+21] and MBPP [AON+21]. We employ 0-shot setting for HumanEval and 3-shot setting for MBPP following prior art [CTJ+21, LXZ+23].

**BIG-Bench Hard** The instruction dataset we generated covers many disciplines, which can potentially enhance the reasoning ability of GLAN. Therefore, we evaluate GLAN on the BIG-Bench Hard dataset (BBH [SSS+22]), which contains 23 challenging tasks from Big-Bench [SRR+23] to assess general reasoning capabilities of LLMs. We employ the standard 3-shot setting with chain-of-thought demonstrations.

**Academic Exams** We also evaluate GLAN on different academic benchmarks to verify whether GLAN is capable of solving exam questions. We choose two benchmarks (i.e., ARC [CCE+18] and MMLU [HBB+20]). Both benchmarks are composed of multi-choice questions. AI2 Reasoning Challenge (ARC [CCE+18]) contains grade-school level, multi-choice science questions. To accurately answer these, a model is expected to not only grasp the underlying knowledge but also poss a certain level of reasoning ability. It contains two sub-sets, which are ARC-Challenge (ARC-C) and ARC-Easy (ARC-E). Massive Multitask Language Understanding (MMLU [HBB+20]) consists of a set of multiple-choice questions about 57 subjects ranging in difficulty from elementary levels to professional levels. It covers various of domains of knowledge, including humanities, STEM and social sciences. Note that there is a training set for ARC. However, we have excluded it from our training set during the decontamination process described in Section 3.1. Previous models mostly leverage probability based methods on ARC and MMLU, which returns the best option based the probabilities of the four options conditioned on the corresponding multi-choice question. We observe in our experiments that after training on 10 million homework questions, GLAN is able to *generate* its predicted options and analysis of multi-choice questions in plain text as `GPT-3.5-turbo` does. We therefore opt for 0-shot setting for GLAN and extract predictions using rules based on its completions as in [MDCM+23].

**Results** Our main results are shown in Table 1. We compare GLAN against general domain models (Orca 2 [MDCM+23], Mistral Instruct [JSM+23] and WizardLM [XSZ+23]), math optimized models (MetaMath [YJS+23] and WizardMath [LSX+23]) and coding optimized models (CodeAlpaca [Cha23]). We also report results of base LLMs (i.e., LLaMA2 [TMS+23] and Mistral [JSM+23]) as references. GLAN either obtains best results or results close to the best across all benchmarks. We observe that capabilities of math or coding optimized models increase on math or coding benchmarks while usually not others. After instruction tuning, GLAN excels on multiple dimensions from mathematical reasoning, coding, reasoning and academic exams with a systematical data generation approach. Also note that our method does not use any task specific training data such as training sets of GSM8K, MATH or ARC as in Orca 2, MetaMath and WizardMath, which indicates the general applicability of GLAN.

**A Closer Look at Academic Exams** ARC and MMLU are all multi-choice based benchmarks on academic exams. However, we observe that improvements of GLAN over Mistral on ARC are much larger than these on MMLU (see Table 1). By grouping the 57 subjects in MMLU to four categories (i.e., STEM, Humanities, Social Sciences and Other (business, health, misc.)), we observe GLAN wildly improves on STEM in MMLU while not other categories (Table 2). Also note that

ARC is composed of high school science problems, which are also STEM questions. GLAN is good at STEM subjects may because responses of our dataset are from `GPT-3.5-turbo`, which by default generates responses with Chain-of-Thoughts (CoT) reasoning. Indeed, we observe that GLAN generates solutions with CoT for multi-choice questions. CoT may help the multi-step reasoning in STEM multi-choice questions [WWS+22], while humanities and social sciences questions involve more with memorization and single step reasoning, where CoT may introduce additional errors.

### 3.4 Task-specific Training Data

GLAN is a generalized method to create synthetic data for instruction tuning. In order to evaluate the generalization capabilities of this synthetic data, we deliberately exclude task-specific training sets from all benchmarks on which we conduct our assessments. Similar to [WZZ+23], we explore whether models have been trained on task specific in-domain data. We compute the training loss $L_{train}$ and test loss $L_{test}$ on ARC Challenge (ARC-C), ARC Easy (ARC-E), GSM8K and MATH for GLAN and other models in comparison. We choose these four datasets because among all benchmarks evaluated in Section 3.3, these benchmarks contain training sets. Intuitively, the larger $\Delta = L_{test} - L_{train}$ is, the more likely the training set is exposed. To make $\Delta$ easier to be interpreted, we additional compute the relative difference $\Delta(\%) = (L_{test} - L_{train})/L_{test}$. Table 3 shows the losses of the training and test splits for GLAN are nearly identical (or $\Delta$ is negative). This suggests that GLAN has not been exposed to in-domain data during training and tuning procedures. Additionally, we observe that GLAN obtains higher losses on both test and training splits on GSM8K, MATH and ARC compared to other models, while results of GLAN on these four datasets are high (see Table 1). This might imply that synthetic data generated by GLAN is diverse and our resulting model avoids convergence to any specific domain or style present in existing benchmarks.

| **Benchmark/Loss** | | **LLaMA2-7B** | **Orca2-7B** | **Mistral-7B-Instruct** | **WizardLM-13B-V1.2** | **GLAN-7B** |
|---|---|---|---|---|---|---|
| **ARC-C** | $L_{test}$ | 2.02 | 2.39 | 2.32 | 2.11 | 4.03 |
| | $L_{train}$ | 2.03 | 2.34 | 2.33 | 2.12 | 4.06 |
| | $\Delta$ | -0.01 | 0.05 | -0.01 | -0.01 | -0.03 |
| | $\Delta$ (%) | **-0.5%** | 2.10% | **-0.43%** | **-0.47%** | **-0.74%** |
| **ARC-E** | $L_{test}$ | 2.10 | 2.47 | 2.51 | 2.18 | 4.31 |
| | $L_{train}$ | 2.12 | 2.43 | 2.54 | 2.20 | 4.32 |
| | $\Delta$ | -0.02 | 0.04 | -0.03 | -0.02 | -0.01 |
| | $\Delta$ (%) | **-0.95%** | 1.61% | **-1.19%** | **-0.91%** | **-0.23%** |
| **GSM8K** | $L_{test}$ | 1.38 | 1.14 | 1.26 | 1.14 | 2.17 |
| | $L_{train}$ | 1.38 | 1.01 | 1.26 | 1.09 | 2.15 |
| | $\Delta$ | 0 | 0.13 | 0 | 0.05 | 0.02 |
| | $\Delta$ (%) | **0%** | 11.4% | **0%** | 4.39% | **0.92%** |
| **MATH** | $L_{test}$ | 1.11 | 1.18 | 1.12 | 1.22 | 1.67 |
| | $L_{train}$ | 1.14 | 1.15 | 1.15 | 1.24 | 1.70 |
| | $\Delta$ | -0.03 | 0.03 | -0.03 | -0.02 | -0.03 |
| | $\Delta$ (%) | **-2.70%** | 2.54% | **-2.67%** | **-1.63%** | **-1.79%** |

Table 3: The evaluation of loss values between the test data and training data. Large positive $\Delta$ (or $\Delta(\%)$) may indicate task specific in-domain training data is exposed to the model during training.

### 3.5 Instruction Following Evaluation

**IFEval** We assess the instruction-following capabilties of GLAN utilizing the Instruction Following Evaluation dataset (IFEval [ZLM+23]). IFEval consists of a collection of "verifiable instructions", encompassing 25 distinct types of instructions (around 500 prompts in total). Each prompt comprises one or more verifiable instructions. The evaluation involves four types of metrics at both prompt-level and instruction-level, evaluating strict and loose accuracies.

As shown in Table 4, GLAN demonstrates superior instruction-following capabilities in both prompt-level and instruction-level evaluations. However, there is still a considerable gap compared to `GPT-3.5-turbo` and `GPT-4`.

**Evol-Instruct Test** Evol-Instruct testset [XSZ+23] contains real-world human instructions from diverse sources and it consists of 218 instances with 29 distinct skills. Each instruction is associated with a difficulty level from 1 to 10. The responses are often open ended descriptions and we believe this benchmark is a necessary supplement to IFEval (answers to their instructions are "verifiable").

| Model | Prompt-level strict-accuracy | Instruction-level strict-accuracy | Prompt-level strict-accuracy | Instruction-level loose-accuracy |
|---|---|---|---|---|
| GPT-3.5-turbo | 53.8 | 64.7 | 56.6 | 67.5 |
| GPT-4 | 77.1 | 83.7 | 79.7 | 85.6 |
| LLaMA2-7B | 14.8 | 27.1 | 16.6 | 29.4 |
| Orca2-7B | 19.4 | 28.9 | 26.1 | 34.7 |
| Mistral-7B-Instruct-v0.1 | 32.0 | 42.8 | 37.7 | 48.0 |
| WizardLM-13B-V1.2 | 23.1 | 33.5 | 26.6 | 37.6 |
| GLAN-7B | **34.0** | **44.8** | **41.2** | **51.6** |

Table 4: Instruction following capability evaluation on IFEval.

Following [XSZ+23] and [CLL+23], we adopt a GPT-4-based automatic evaluation method to conduct a pairwise comparison between GLAN and other models. Specifically, GPT-4 is instructed to assign a score between 1 and 10 overall score w.r.t. the helpfulness, relevance, accuracy, and level of detail of responses generated by two different models for a given input question. A higher score indicates better overall performance. To mitigate potential order bias, we perform bidirectional comparisons for each response pair and determine their average score. The average score difference to GLAN (i.e., `avg_score(GLAN)` $-$ `avg_score`$(x)$) serves as the final metric. Table 5 presents the results of pairwise comparisons across various levels of instruction difficulty. GLAN showcases superior performance compared to LLaMA-2, Orca 2, Mistral Instruct, and even WizardLM-13B (note that GLAN contains only 7B parameters) on most difficulty levels and overall scores. This suggests that GLAN demonstrates improved ability to process diverse instructions, regardless of their difficulty or complexity. Also note that GLAN falls behind `GPT-3.5-turbo` as other models in comparison. Additionally, we group Evol-Instruct test according to the 29 skills and we observe the same trends. Detailed results are in Appendix (Table 7). GLAN demonstrates strong performance on most skills especially on Math, Coding and Reasoning. However, it slightly falls short in common-sense related tasks.

| Difficulty | Ratio | LLaMA2-7B | Orca2-7B | Mistral-7B-Instruct | Wizard-13B-V1.2 | GPT-3.5-turbo |
|---|---|---|---|---|---|---|
| 1 | 5.1% | 5.41 | 2.23 | -0.37 | -0.21 | -2.41 |
| 2 | 8.7% | 5.87 | 1.74 | 1.06 | 1.41 | -1.18 |
| 3 | 12.4% | 5.72 | 2.35 | 1.04 | 1.37 | -1.14 |
| 4 | 10.5% | 5.61 | 1.34 | 1.52 | 1.54 | -0.92 |
| 5 | 4.1% | 4.67 | 3.31 | 2.39 | 2.5 | -0.45 |
| 6 | 19.3% | 4.43 | 2.42 | 0.74 | 1.54 | -1.36 |
| 7 | 11.0% | 4.97 | 1.26 | 1.62 | 1.36 | -0.41 |
| 8 | 17.9% | 6.02 | 3.58 | 3.17 | 1.7 | 0.15 |
| 9 | 6.0% | 6.35 | 4.2 | 1.36 | 0.9 | -0.92 |
| 10 | 5.1% | 5.14 | -0.05 | 1.53 | -0.54 | -0.85 |
| (1-5) Easy | 41.00% | **5.46** | **2.19** | **1.13** | **1.32** | -1.22 |
| (6-10) Hard | 59.00% | **5.38** | **2.28** | **1.68** | **0.99** | -0.68 |

Table 5: Pairwise comparison on various difficulty levels between GLAN and other models on Evol-Instruct testset. The scores are the average gap of scores assigned by GPT-4, calculated as `avg_score(GLAN)` $-$ `avg_score`$(x)$.

**GLAN-Test** There are only hundreds of instructions in In IFEval and Evol-Instruct Test and we believe the domains or skills they can cover are rather limited. Therefore, we propose a held-out test set using GLAN data and we call it GLAN-Test. It contains 6,300 instructions on 126 disciplines (50 instructions for each discipline). We further categorize the 126 disciplines to 8 distinct *fields* (i.e., Academic-Humanities, Academic-Social Science, Academic-Natural Science, Academic-Applied Science, Academic-Formal Science, Industry-Manufacturing, Industry-Services and Industry-Agriculture). We believe that the extensive domain coverage of GLAN-Test renders it an effective test bed for the assessment of generalization capabilities in LLMs. We adopt the same GPT-4 based evaluation protocol as in Evol-Instruct Test (previous paragraph). We prompt GPT-4 to do a pairwise ranking of GLAN and other models in comparison. The overall results and results across the 8 fields are presented in Table 6, where GLAN obtains higher GPT-4 scores than Orca2-7B, Mistral-7B Instruct and WizardLM-13B, despite using only 7B parameters. GLAN still lag behind `GPT-4`. Detailed results for the 126 fine-grained disciplines can be found in Appendix A.2 (see Table 8 for more details). GLAN demonstrates its effectiveness on multiple domains (or disciplines) such as Mathematics, Physics, Chemistry, Computer science, Electrical, Mechanical, etc., indicating that smaller models may yield general improvements on various domains through strategic fine-tuning. Furthermore, it is noted that GLAN demonstrates less-than-ideal performance across distinct disciplines such as American history, Divinity, or Radiology. This observation underscores the potential for further refinement and development of our methodology within these domains.

| **Field (Ratio)** | **Orca2-7B** | **Mistral-7B-Instruct** | **WizardLM-13B-V1.2** | **GPT-4** |
|---|---|---|---|---|
| Academic-Humanities (15.9%) | 0.79 | 0.25 | 0.02 | -0.62 |
| Academic-Social Science (7.9%) | 1.22 | 0.21 | 0.09 | -0.63 |
| Academic-Natural Science (4.0%) | 1.73 | 1.23 | 0.53 | -0.5 |
| Academic-Applied Science (42.1%) | 1.58 | 0.32 | 0.08 | -0.58 |
| Academic-Formal Science (3.2%) | 3.87 | 2.48 | 2.32 | -0.55 |
| Industry-Manufacturing (12.7%) | 2.26 | 0.56 | 0.33 | -0.43 |
| Industry-Services (11.9%) | 1.82 | 0.23 | 0.09 | -0.5 |
| Industry-Agriculture (2.4%) | 1.2 | 0.46 | 0.13 | -0.33 |
| Overall (100.0%) | **1.61** | **0.43** | **0.19** | -0.55 |

Table 6: Pairwise comparison between GLAN and other models on GLAN-Test (the 126 disciplines are categorized into 8 fields for clarity of the illustration). The scores are the average gap of scores assigned by GPT-4, calculated as $\texttt{avg\_score(GLAN)} - \texttt{avg\_score}(x)$.

## 4 Related Work

Recent literature has extensively explored the collection of various human-made resources for instruction tuning. An intuitive direction is to collect existing NLP datasets and corresponding task descriptions [SWR+22, WMA+22, ZLX+23], typical LLMs such as BLOOMZ [MWS+22] and FLAN [WBZ+21] are trained on this type of instruction tuning data. However, with only tens to thousands of existing datasets available, the scope and diversity of instruction tuning are inevitably limited. Another common practice is to implement instruction tuning with real-world human user prompts. For instance, InstructGPT [OWJ+22] was trained on high-quality human prompts submitted by real-world users to OpenAI GPT APIs. Vicuna [CLL+23] leverages user-shared prompts along with ChatGPT responses for instruction tuning, and Dolly[CHM+23] was trained on simulated human-user interactions written by over 5k employees. Nevertheless, acquiring instructional data from human users typically involves high costs and involves privacy concerns.

As LLM capabilities improve, instruction tuning with LLM-generated data exhibits better scalability and potential in addressing the super-alignment problem [SJH+23]. Leveraging the in-context learning ability of LLMs, Unnatural instructions [HSLS22] and Self-instruct [WKM+22] sampled seed instructions as examples to elicit LLMs to generate new instructions. Taking advantage of the rephrasing ability of LLMs, WizardLM [XSZ+23] and WizardMath [LSX+23] were trained using Evol-Instruct. Evol-Instruct iteratively employs ChatGPT to rewrite seed instructions into increasingly complex instructions. Similar to generation from seed instructions, carefully selected seed topics are used for generating textbook-like synthetic data [LBE+23] or self-chat multi-turn dialogues [XGDM23, DCX+23] for instruction tuning. However, models trained on these LLM-generated data only work well in specific domains such as math [LSX+23, YJS+23], dialogue [XGDM23, DCX+23] or open-ended question answering [TGZ+23, XSZ+23]. These methods encounter challenges in generalization [GWS+23], as the data diversity is restricted by seed instructions or seed topics.

## 5 Conclusions

We propose GLAN, a general and scalable method for synthesizing instruction data. Experiments show GLAN can help large language models improve their capabilities in multiple dimensions from mathematical reasoning, coding, academic exams, logical reasoning to general instruction following. Currently, our synthetic data is based on the taxonomy of human knowledge and capabilities and there are other types of useful data not been covered. We are interested to design methods with boarder coverage. Our current instruction data are mostly question answer pairs and in the next step, we plan to generate synthetic data of multi-turn conversations and long documents.

## References

[AON+21] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[CCE+18] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[Cha23] Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. `https://github.com/sahil280114/codealpaca`, 2023.

[CHM+23] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023.

[CKB+21] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[CLL+23] 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. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[CTJ+21] 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. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[DCX+23] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

[GWS+23] Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. The false promise of imitating proprietary llms. *arXiv preprint arXiv:2305.15717*, 2023.

[HBB+20] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[HBD+19] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[HBK+21] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[HBM+22] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[HSLS22] Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *ArXiv*, abs/2212.09689, 2022.

[JSM$^+$23] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[KMH$^+$20] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[LBE$^+$23] Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*, 2023.

[LHV$^+$23] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*, 2023.

[LSX$^+$23] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

[LXZ$^+$23] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*, 2023.

[MDCM$^+$23] Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, et al. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*, 2023.

[MMJ$^+$23] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*, 2023.

[MWS$^+$22] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*, 2022.

[OWJ$^+$22] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[SJH$^+$23] Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. Large language model alignment: A survey. *arXiv preprint arXiv:2309.15025*, 2023.

[SRR$^+$23] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick

Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Batchelder, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mitch Walker, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan A. Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima

Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima, Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023.

[SSS+22] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[SWR+22] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*, 2022.

[TGZ+23] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[TMS+23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[WBZ+21] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[wik23] Education, 2023. Last edited on 24 March 2023.

[WKM+22] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

[WMA+22] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*, 2022.

[WWS+22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[WZZ+23] Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. Skywork: A more open bilingual foundation model, 2023.

[XGDM23] Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.

[XSZ+23] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

[YJS+23] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

[ZLM+23] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[ZLX+23] Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*, 2023.

# A Appendix

## A.1 Evol-Instruct Test Results on Different Skills

| **Skill** | **Ratio** | **LLaMA2-7B** | **Orca2-7B** | **Mistral-7B-Instruct** | **Wizard-13B-V1.2** | **GPT-3.5-turbo** |
|---|---|---|---|---|---|---|
| Math | 8.7% | 6.58 | 2.16 | 2.41 | 2.46 | -1.42 |
| Code Generation | 8.3% | 6.16 | 3.87 | 4.22 | 2.59 | -0.25 |
| Writting | 8.3% | 5.2 | 0.79 | -0.22 | 0.24 | -1.1 |
| Computer Science | 6.9% | 7.1 | 4.4 | 0.83 | 1.22 | 0.02 |
| Reasoning | 6.0% | 6.3 | 2.52 | 3.38 | 3.02 | 0.62 |
| Complex Format | 5.5% | 3.13 | 3.5 | -0.17 | 2.41 | -1.96 |
| Code Debug | 4.6% | 5.85 | 2.3 | 1.4 | 0.2 | -2.5 |
| Common-Sense | 4.1% | 6.5 | 3.19 | -1.33 | -0.92 | -2.78 |
| Counterfactual | 3.7% | 7.06 | 2.15 | 3 | 1.5 | 0.72 |
| Multilingual | 3.2% | 7.35 | 0.79 | 1.71 | -0.68 | -2.75 |
| Roleplay | 2.8% | 7.08 | 2.25 | 3.5 | 0.92 | -0.59 |
| Biology | 2.8% | 6.66 | 2.75 | 1.46 | -0.09 | 1.38 |
| Technology | 2.8% | -0.08 | 2.54 | -3 | -1.5 | -2.75 |
| Ethics | 2.8% | 6.59 | 3.38 | 2.41 | 5.42 | -0.21 |
| TruthfulQA | 2.3% | 3.1 | 3.7 | -1.05 | -1.3 | -0.85 |
| Sport | 2.3% | 4.3 | 0.55 | -0.2 | 4.8 | -0.3 |
| Law | 2.3% | 7.7 | 4.65 | 5.85 | 1.7 | 0.2 |
| Medicine | 2.3% | 3.9 | -2.05 | 1.9 | 0.15 | -1.25 |
| Literature | 2.3% | 6.3 | 1.9 | 0.2 | 1.45 | -0.15 |
| Entertainment | 2.3% | 4.5 | 2.7 | -3 | 1.9 | -3.2 |
| Art | 2.3% | 4.9 | 1 | 2.9 | -0.85 | -2.05 |
| Music | 2.3% | 4.4 | 4.1 | 0.5 | 1.45 | -2.3 |
| Toxicity | 1.8% | 7.25 | 3.12 | 3.75 | 1.63 | -1.32 |
| Economy | 2.3% | 6 | 0.15 | 1.9 | 0 | 0 |
| Physics | 2.3% | 6.8 | 2.5 | 4.35 | 3.65 | -1 |
| History | 1.8% | 4.12 | -0.56 | 3.76 | -0.31 | 0.12 |
| Academic Writing | 1.8% | 6.76 | 6.37 | 2.44 | 1.37 | 0.62 |
| Chemistry | 0.9% | 9.5 | 0.63 | 5.25 | 2.5 | 0.75 |
| Philosophy | 0.5% | 11 | -0.25 | 0.25 | -0.25 | 0.5 |
| Avg.(29 skills) | 100% | 5.42 | 2.24 | 1.41 | 1.16 | -0.95 |

Table 7: Pairwise comparison on various skills between GLAN and other models on Evol-Instruct testset. The scores are the average gap of scores assigned by GPT-4, calculated as $\texttt{avg\_score(GLAN)} - \texttt{avg\_score}(x)$.

## A.2 GLAN-Test Results on Different Disciplines

| **Discipline** | **Orca-2-7b** | **Mistral-7B-Instruct-v0.1** | **WizardLM-13B-V1.2** | **GPT-4** |
|---|---:|---:|---:|---:|
| Avg. | 1.61 | 0.43 | 0.19 | -0.55 |
| Advertising | 1.92 | 0.46 | 0.21 | -0.04 |
| Aerospace industry | 3.24 | 1.24 | 0.6 | -0.42 |
| Agriculture | 2.44 | 0.04 | -0.05 | -0.48 |
| American history | -0.49 | -0.27 | -0.76 | -0.83 |
| American politics | 1.23 | -0.3 | -0.4 | -0.87 |
| Anthropology | 0.59 | 0.17 | 0.06 | -0.27 |
| Applied mathematics | 3.75 | 2.6 | 2.74 | -0.47 |
| Archaeology | 2.59 | -0.11 | 0.1 | -0.56 |
| Architecture and design | 2.63 | 0.34 | 0.4 | -0.37 |
| Astronomy | 1.01 | 0.83 | 0.03 | -0.44 |
| Automotive industry | 1.27 | 0.71 | 0.46 | -0.06 |
| Biblical studies | -0.05 | 0.33 | -0.47 | -0.65 |
| Biology | 1.09 | 0.22 | -0.09 | -0.17 |
| Business | 3.61 | 1.14 | 0.88 | -0.26 |
| Chemical Engineering | 3.15 | 1.6 | 1.18 | -0.77 |
| Chemistry | 3.06 | 2.09 | 0.8 | -0.87 |
| Civil Engineering | 1.94 | 0.74 | 0.75 | -0.25 |
| Clinical laboratory sciences | 1.32 | 0.94 | -0.11 | -0.47 |
| Clinical neuropsychology | 2.15 | 0.29 | 0.25 | -0.4 |
| Clinical physiology | 2.07 | 0.41 | 0.51 | -0.08 |
| Communication studies | 0.3 | 0.26 | -0.15 | -0.3 |
| Computer science | 4.29 | 1.45 | 1.9 | -0.33 |
| Cultural industry | 3.15 | 0.44 | 0.05 | -0.36 |
| Dance | 2.11 | 0.21 | 0.4 | -0.47 |
| Dentistry | 1.67 | 0.66 | 0.48 | 0.01 |
| Dermatology | 2.12 | 0.55 | -0.05 | -0.65 |
| Divinity | -0.34 | -0.17 | -0.48 | -0.89 |
| Earth science | 0.39 | 0.44 | -0.08 | -0.33 |
| Economics | 2.62 | 0.96 | 0.62 | -0.4 |
| Education | 2.67 | 0.42 | 0.2 | -0.84 |
| Education industry | 2.19 | 0.4 | 0.56 | -1.33 |
| Electric power industry | 3.23 | 1.31 | 0.39 | -0.79 |
| Electrical Engineering | 3.81 | 1.26 | 1.41 | -0.34 |
| Emergency medicine | 2.04 | 0.44 | -0.18 | -0.86 |
| Energy industry | 3.59 | 0.98 | 0.54 | -0.22 |
| Environmental studies and forestry | 0.12 | 0.41 | 0.1 | -0.45 |
| Epidemiology | 3.02 | 0.52 | 0.33 | -0.46 |
| European history | 0.14 | 0.62 | 0.15 | -0.18 |
| Fashion | 2.5 | 0.66 | 0.47 | -0.53 |
| Film | 0.76 | 0.45 | -0.16 | -0.78 |
| Film industry | 1.58 | 0.46 | 0.25 | -0.59 |
| Fishing industry | 1.67 | 1 | 0.57 | -0.09 |
| Floral | 1.92 | 0.89 | 0.58 | -0.09 |
| Food industry | 3.64 | 0.12 | 0.14 | -0.42 |
| Foreign policy | 2.4 | 0.49 | 0.16 | -0.46 |
| Geography | 0.88 | 0.6 | 0.28 | -0.66 |
| Geriatrics | 2.19 | -0.32 | -0.56 | -0.71 |
| Gynaecology | 1.05 | -0.27 | -0.26 | -0.67 |
| Healthcare industry | 1.62 | -0.25 | 0.14 | -0.5 |
| Hematology | 0.35 | 0.32 | -0.05 | -0.72 |
| History | 0.75 | 0.54 | -0.04 | -0.38 |
| Holistic medicine | 0.85 | 0.48 | 0.26 | -0.27 |
| Hospitality industry | 2.36 | 0.48 | 0.28 | -0.07 |
| Housing | 4.04 | 0.15 | -0.22 | -0.62 |
| Industrial robot industry | 3.84 | 1.22 | 0.84 | -0.71 |
| Infectious disease | 1.76 | 0.14 | 0.18 | -0.56 |
| Insurance industry | 2.67 | 0.42 | 0.61 | -0.4 |
| Intensive care medicine | 1.11 | 0.56 | 0.08 | -0.33 |
| Internal medicine | 1.02 | 0.45 | -0.01 | -0.42 |
| Journalism | 2.77 | -0.13 | -0.21 | -0.69 |
| Languages and literature | 0.45 | 0.05 | -0.39 | -0.84 |
| Law | 0.42 | 0.39 | 0.04 | -0.49 |
| Leisure industry | 1.49 | 0.12 | -0.09 | -0.49 |
| Library and museum studies | 1.52 | 0.5 | 0.33 | -0.32 |

| **Discipline** | **Orca-2-7b** | **Mistral-7B-Instruct-v0.1** | **WizardLM-13B-V1.2** | **GPT-4** |
|---|---:|---:|---:|---:|
| Linguistics | 0.39 | 0.38 | -0.12 | -0.96 |
| Logic | 2.95 | 1.56 | 1.62 | -0.79 |
| Materials Science and Engineering | 1.71 | 0.97 | 0.54 | -0.91 |
| Mathematics | 4.69 | 3.81 | 2.73 | -0.61 |
| Mechanical Engineering | 2.25 | 1.71 | 1.15 | -0.95 |
| Medical toxicology | 0.62 | 0 | 0.11 | -1.01 |
| Medicine | 1.49 | 0.93 | 0.36 | -0.37 |
| Military sciences | 0.42 | 0.53 | 0.17 | -0.45 |
| Mining | 3.17 | 0.32 | 0.41 | -0.61 |
| Music | 2.85 | 0.38 | 1.07 | -0.05 |
| Music industry | 2.05 | -0.03 | -0.08 | -0.8 |
| Nursing | 1.49 | 0.14 | -0.12 | -0.59 |
| Nutrition | 1.15 | -0.2 | -0.13 | -0.65 |
| Obstetrics | 1.49 | 0.08 | -0.43 | -0.53 |
| Ophthalmology | 0.97 | 0.01 | -0.47 | -0.97 |
| Otolaryngology | 1.51 | -0.44 | -0.29 | -1.11 |
| Pathology | 0.23 | 0.35 | 0.19 | -0.72 |
| Pediatrics | 1.62 | 0.55 | -0.34 | -0.47 |
| Performing arts | 0.38 | 0.09 | -0.36 | -1.06 |
| Petroleum industry | 3.12 | 0.44 | 0.08 | -0.54 |
| Pharmaceutical industry | 2.75 | 0.41 | 0.4 | -0.46 |
| Pharmaceutical sciences | 0.77 | 0.19 | 0.16 | -0.8 |
| Philosophy | 0.51 | 0.25 | 0.49 | -0.64 |
| Physics | 3.15 | 2.67 | 2.05 | -0.73 |
| Political science | 0.04 | -0.05 | -0.31 | -0.91 |
| Prehistory | 0.35 | 0.19 | 0.05 | -0.41 |
| Preventive medicine | 2.69 | 0.57 | 0.09 | -0.36 |
| Psychiatry | 2.93 | 0.27 | -0.07 | -0.32 |
| Psychology | 0.53 | -0.02 | -0.3 | -0.96 |
| Public administration | 0.94 | -0.27 | 0.1 | -1.2 |
| Public health | 1.21 | 0.07 | 0.22 | -0.56 |
| Public policy | 0.78 | -0.06 | -0.28 | -0.92 |
| Pulp and paper industry | 1.13 | 0.63 | 0.57 | -0.25 |
| Radiology | -0.17 | -0.19 | -0.82 | -0.62 |
| Real estate industry | 1.01 | 0.02 | -0.12 | -0.5 |
| Religious Studies | 0.38 | 0 | -0.32 | -0.63 |
| Retail industry | 1.1 | -0.25 | -0.37 | -0.6 |
| Semiconductor industry | 1.49 | 0.64 | 0.71 | -0.42 |
| Sexology | 1.81 | -0.44 | -0.37 | -0.96 |
| Shipbuilding industry | 1.54 | 0.37 | 0.42 | -0.32 |
| Social work | 0.93 | -0.42 | -0.53 | -0.77 |
| Sociology | 1.49 | 0.21 | 0.76 | -0.3 |
| Steel industry | 0.88 | 0.45 | 0.09 | -0.34 |
| Surgery | 0.86 | -0.02 | -0.35 | -0.73 |
| Systems science | 1.9 | 0.56 | 0.41 | -0.45 |
| Telecommunications industry | 1.81 | 0.4 | 0.39 | -0.27 |
| Television | 0.37 | -0.33 | -0.69 | -1 |
| Textile industry | 0.82 | -0.26 | -0.68 | -0.59 |
| Theatre | 0.31 | -0.27 | -0.34 | -1.07 |
| Theology | -0.38 | 0.37 | -0.45 | -0.54 |
| Tobacco industry | 0.59 | -0.13 | -0.48 | -0.67 |
| Transport industry | 1.19 | -0.33 | -0.36 | -0.56 |
| Transportation | 1.74 | 0.26 | 0.17 | -0.74 |
| Urology | 0.05 | -0.29 | -0.36 | -0.64 |
| Veterinary medicine | -0.14 | 0.36 | -0.31 | -0.62 |
| Video game industry | 1.67 | 0.2 | -0.24 | -0.62 |
| Visual arts | 0.98 | 0.22 | 0.26 | -0.56 |
| Water industry | 0.9 | -0.11 | -0.09 | -0.51 |
| Wood industry | 1.36 | 0.5 | 0.31 | -0.25 |

Table 8: Pairwise comparison across 126 disciplines (or domains) on *GLAN-Test*. The scores are generated from the average gap between GLAN and other model $x$ in assessment scores assigned by GPT-4, calculated as $\texttt{avg\_score(GLAN)} - \texttt{avg\_score}(x)$.