LLMs as Workers in Human-Computational Algorithms?
Replicating Crowdsourcing Pipelines with LLMs

Tongshuang Wu ∗
Haiyi Zhu
Maya Albayrak
Alexis Axon
Amanda Bertsch
Wenxing Deng
Ziqi Ding
Bill Guo
Sireesh Gururaja
Tzu-Sheng Kuo
Jenny T. Liang
Ryan Liu
Ihita Mandal
Jeremiah Milbauer
Xiaolin Ni
Namrata Padmanabhan
Subhashini Ramkumar
Alexis Sudjianto
Jordan Taylor
Ying-Jui Tseng
Patricia Vaidos
Zhijin Wu
Wei Wu
Chenyang Yang
Carnegie Mellon University, Pittsburgh, PA, USA
sherryw@cs.cmu.edu

Abstract

LLMs have shown promise in replicating human-like behavior in crowdsourcing tasks that were previously thought to be exclusive to human abilities. However, current efforts focus mainly on simple atomic tasks. We explore whether LLMs can replicate more complex crowdsourcing pipelines. We find that modern LLMs can simulate some of crowd-workers’ abilities in these “human computation algorithms,” but the level of success is variable and influenced by requesters’ understanding of LLM capabilities, the specific skills required for sub-tasks, and the optimal interaction modality for performing these sub-tasks. We reflect on human and LLMs’ different sensitivities to instructions, stress the importance of enabling human-facing safeguards for LLMs, and discuss the potential of training humans and LLMs with complementary skill sets. Crucially, we show that replicating crowdsourcing pipelines offers a valuable platform to investigate (1) the relative strengths of LLMs on different tasks (by cross-comparing their performances on sub-tasks) and (2) LLMs’ potential in complex tasks, where they can complete part of the tasks while leaving others to humans.

1 Introduction

The rapid advancement of AI systems has revolutionized our understanding of the capabilities of machines. One remarkable breakthrough in this field is the emergence of Large Language Models (LLMs, e.g., ChatGPT). With a combination of extensive pre-training (Brown et al., 2020) and instruction tuning (Stiennon et al., 2020; Wu et al., 2023; Ziegler et al., 2019), LLMs now not only possess a large amount of world knowledge, but can effectively leverage this knowledge to accomplish various tasks simply by following instructions.

Various studies have reported that these models can replicate human-like behavior to some extent, which is a key objective in the training of AI models (Wang et al., 2022a; Bubeck et al., 2023). In particular, a large proportion of these studies have been using LLMs to replicate crowdsourcing tasks, possibly because they represent a wide range of tasks that were previously considered exclusive to human computational capabilities (Bernstein, 2013). For example, LLMs can generate annotations of higher quality at a reduced cost compared to crowd-workers or even experts (Gilardi et al., 2023; Törnberg, 2023; Ziegler et al., 2019), and can approximate human opinions in subjective tasks, allowing for simulated human responses to crowdsourced questionnaires and interviews (Hämäläinen et al., 2023; Argyle et al., 2022). These observations indicate that LLMs will have significant social and economic implications, potentially reshaping the workforce by replacing certain human jobs (Eloundou et al., 2023). In fact, some studies have observed that now crowd-workers tend to rely on LLMs for completing text production tasks (Veselovsky et al., 2023).

However, most existing efforts tend to focus on atomic tasks that are simple, self-contained, and easy for a single crowdworker to complete in a
short amount of time — the most basic version of human computational power. These efforts also are scattered across various tasks and domains, making it hard to systematically compare and understand which tasks LLMs may excel or underperform at, and to what extent they can simulate, replace, or augment humans on specific tasks. Such emphases prompt us to ask, how far does the LLM replicability generalize? Will they be useful in more advanced formats of “human computation”?

We are especially interested in whether LLMs can be used to replicate crowdsourcing pipelines, which represent a more sophisticated approach to harnessing human computation (Little et al., 2010). In a typical pipeline, complex tasks are broken down into pieces (sub-tasks) that can be performed independently, then later combined (Chilton et al., 2013; Kim et al., 2017; Law and Zhang, 2011; Retelny et al., 2017). This method has been widely used to scale crowdsourcing usability, allowing it to handle tasks that are too challenging for individual crowdworkers with limited level of commitment and unknown expertise (e.g., summarizing lengthy novels, software development, or deciphering heavily blurred text; Kittur et al., 2011).

Interestingly, research on LLMs has also explored scaling their capabilities for more complex tasks through chaining. Though named differently, LLM chains and crowdsourcing pipelines share similar motivation and strategy of scaling LLM utility. Previous studies have connected the two, noting that they decompose tasks to address different problems (Wu et al., 2022b): crowdsourcing pipelines focus on factors affecting human worker performance, such as cognitive load and task duration, while LLM chains address inherent limitations of LLMs, such as high variance in prompt effectiveness. However, since LLMs have now been trained to better align with humans in following instructions and handling complex contexts (Ouyang et al., 2022), it is possible for human and LLM workers to adopt the same task division strategies.

In this study, we investigate the potential of LLMs to replace human workers in advanced human computation processes. To accomplish this, we designed a course assignment for a special topic course named Human-Centered NLP at Carnegie Mellon University. In the assignment, 20 students were tasked to select one (out of seven) crowdsourcing pipelines depicted in prior work, and replicate them by employing LLMs to handle each sub-task. The replication study also offers an interesting bonus analysis point: While LLM modules in a chain perform unique sub-tasks, all the sub-tasks occur in the same application domain (e.g., processing the same document in different ways), making it fairer to compare LLMs’ performance in different sub-tasks and uncovering the relative strengths and weaknesses.

We find that while LMs appear to be able to replicate crowdsourcing pipelines, there is a wide variance in which parts they tend to perform well / in ways we would expect from humans (main findings in Table 2b). The differences emerge from two primary reasons. First, LLMs and humans respond differently to instructions. LLMs are more responsive to adjectives and comparison-based instructions, such as “better” or “more diverse,” whereas humans handle instructions involving trade-off criteria better. Second, humans receive more scaffolds through disagreement resolution mechanisms and interface-enforced interactions, enabling guardrails on output quality and structure that are not available to LLMs. These observations highlight the need to improve LLM instruction tuning to better handle ambiguous or incomplete instructions, as well as the necessity to consider how non-textual “instructions” can be employed either during LLM finetuning or actual usage. Moreover, the effectiveness of replicated LLM chains depends on students’ perceptions of LLM strengths, which calls for more investigations on assisted prompting.

In addition to offering immediate insights into the differences between LLMs and crowdworkers, our research demonstrates that replicating crowdsourcing pipelines serves as a valuable platform for future investigations into the partial effectiveness of LLMs across a wider range of tasks. Rather than expecting LLMs to tackle entire complex tasks, we can instead assess and identify specific sub-tasks in which LLMs consistently perform on par with humans. This evidence can then be utilized to distribute sub-tasks between LLMs and human workers, optimizing the allocation of responsibilities. We opensource the prompt chains, outputs, and evaluation at https://github.com/tongshuangwu/llm-crowdsourcing-pipeline.

2 Background and Related Work

Crowdsourcing helps solve problems that require human inputs, at scale (Howe et al., 2006). Particularly in earlier times when AI capabilities were
limited, crowdsourcing was seen as a promising approach to leverage and enhance the unique computational powers possessed by humans.

A key focus of crowdsourcing research has been the development of pipelines to tackle increasingly complex crowdsourcing goals (Kittur et al., 2011). Through careful task decomposition, crowdsourcing pipelines strategically collect inputs from human workers, capitalizing on their strengths while mitigating their limitations. This feat is challenging, if not impossible, to achieve in traditional crowdsourcing designs. For example, Bernstein et al. (Bernstein et al., 2010) ensured text editing quality through a Find-Fix-Verify workflow, which modulates the scope of sub-tasks to reduce variance of crowdworker effort. Meanwhile, Context Trees (Verroios and Bernstein, 2014) hierarchically summarize and trim the otherwise overwhelming global contexts, making them compact enough for a single worker to digest. Because of their sophisticated designs, crowdsourcing pipelines are often referred to as human computation algorithms or crowd algorithms (Howe et al., 2006; Law and Zhang, 2011; Kittur et al., 2011; Little et al., 2010).

Though emerged in a completely separate field (NLP), LLM Chains share similar goals with crowdsourcing pipelines — to complete complex tasks that are challenging to perform in one pass. This decomposition can take either an explicit or implicit form. For example, Chain-of-Thought (Kojima et al., 2022; Wei et al., 2022) employs prompts like “let’s consider this step-by-step” makes LLMs to resolve sub-tasks that are not pre-defined, whereas AI Chains (Wu et al., 2022b) and Decomposed Prompting (Khot et al., 2022) explicitly define sub-tasks and employ distinct prompts for each sub-task. More recently, opensource libraries like LangChain (Chase) and services like PromptChainer (Wu et al., 2022a; noa) have enabled practitioners to create LLM chains for tackling tasks involving intricate compositionality.

As reviewed in Section 1, Wu et al. (2022b) has drawn explicit connections between LLM chaining and crowdsourcing pipelines. Besides similar motivations, these two methods also share similar challenges, e.g., handling cascading errors that affect later stages (Kittur et al., 2011) or synthesizing workers’ inconsistent contributions (Kittur et al., 2011; Bernstein et al., 2010), but these challenges can be utilized for enhancing the transparency and debuggability of AI-infused systems. More importantly, Wu et al. (2022b) distinguished the task decomposition objectives for the two approaches: for tackling different limitations of humans and LLM workers. While theoretically this assertion remains true, in practice the differences between humans and LLM workers seem to get blurred. With LLMs evolving to process longer context (OpenAI, 2023), following instructions more closely (Ouyang et al., 2022), and exhibiting improved reasoning capability (Bubeck et al., 2023), some of their limitations start to overlap with those of humans. Various recent work also testifies this observation: Although not explicitly categorized as chaining, several studies have employed strategies to have LLMs self-improve in multiple runs, such as self-task (Press et al., 2022), self-reflection (Shinn et al., 2023), and self-consistency (Wang et al., 2022b), some of which are similar to crowdsourcing pipelines. These recent developments of LLMs, and the success of crowdsourcing pipelines, prompt us to re-assess whether the idea of human computation algorithms can be directly transferred to AIs.

3 Study Design

Study Procedure The study required participants (students) to replicate a crowdsourcing pipeline by writing multiple prompts that instruct LLMs to complete different microtasks.

To accomplish this, the students began by thoroughly reading a crowdsourcing pipeline paper for replication. To demonstrate the effectiveness of their replicated pipeline, they were also asked to determine an appropriate testing task, create at least three test cases consisting of pairs of inputs and ideal outputs, and self-propose a set of task-dependent metrics for evaluating pipeline outputs (e.g., fluency, creativity, coherence). Then, they were instructed to implement two solutions: (1) a baseline solution that prompts one LLM module to complete the entire task (Baseline), and (2) a replica of their chosen crowdsourcing pipeline (LLM Chain). They compared the two LLM solutions using their designated test cases and metrics, providing the reasoning behind their ratings. Finally, they concluded the task by reflecting on why the LLM chain replication either succeeded or failed and brainstormed possible ways to improve the chains in the future.

After students submitted their assignments, they underwent a peer-grading process. In this process, each student’s submission was assessed by three of
<table>
<thead>
<tr>
<th>Pipeline</th>
<th>Description</th>
<th>Sample Task</th>
<th>Replication evaluation</th>
</tr>
</thead>
</table>
| **Map-Reduce**               | *Kittur et al., 2011*  
Partition tasks into discrete subtasks, Map subtasks to workers, Reduce / merge their results into a single output | Write essay | Total: 4  | Unique: 1  | Correct: 3  | Effective: 3  |
| **HumorTool**                | *Chilton et al., 2016*  
Define semantic roles as the answers to a series of questions that are intuitive for non-experts. | Create satire | Total: 4  | Unique: 2  | Correct: 3  | Effective: 1  |
| **Iterative Process**        | *Little et al., 2010*  
Feed the result of one creation task into the next, so workers see content generated by previous workers. | Brainstorm  | Total: 3  | Unique: 2  | Correct: 3  | Effective: 2  |
| **Microtasking**             | *Cheng et al., 2015*  
Concrete microtasking for sorting task: an implementation of human-powered quicksort | Sorting  | Total: 3  | Unique: 3  | Correct: 3  | Effective: 1  |
| **Find-Fix-Verify**          | *Bernstein et al., 2010*  
For writing and editing: Find problems, Fix the identified problems, Verify these edits | Shorten text  | Total: 3  | Unique: 3  | Correct: 2  | Effective: 1  |
| **Price-Divide-Solve**       | *Kulkarni et al., 2012*  
Workers recursively divide complex steps until they are at an appropriately simple level, then solve them. | Write essay  | Total: 1  | Unique: 1  | Correct: 1  | Effective: 1  |
| **Task Paraphrase**          | *He et al., 2015*  
Define semantic roles as the answers to a series of questions that are intuitive for non-experts. | SRL labeling  | Total: 1  | Unique: 1  | Correct: 1  | Effective: 1  |

Table 1: Crowdsourcing pipelines replicated, and their example outputs from student-replicated LLM chains.
their peers in a double-blind manner. The peers rated the submissions based on replication correctness, thoroughness, and comprehensiveness of their envisioned LLM chain improvements. They rated all the criteria on a five-level Likert Scale and supplied detailed reasoning for their grading. The instructor carefully reviewed the gradings and excluded any assessments that appeared to lack thoughtful reflections or misunderstood the submissions. The full assignment instruction, the peer grading form, as well as the student submissions are all available at https://github.com/tongshuangwu/llm-crowdsourcing-pipeline.

Participants 21 students (13 females, 8 males) completed the task as one of their assignments for the Spring 2023 course 05-499/899: Human-Centered NLP. This comprised of 6 undergraduates, 10 master’s students, and 5 PhD students specializing in Sociology, Learning Science, Human-Computer Interaction, or Natural Language Processing. The paper presents findings from 20 students’ submissions, as one student opted for a non-programming approach for partial credit.

Crowdsourcing papers We selected crowdsourcing papers based on three criteria: (1) Diversity: the papers should represent different pipeline designs (iterative, parallel), intermediate steps (question-answering, comparison, editing), and tasks (creative tasks, annotation tasks, editing tasks, etc.). (2) Replicability: The papers should provide clear definitions for each sub-step and concrete sample test cases. Considering our emphasis on LLMs, we exclusively considered papers that described tasks with textual inputs and outputs. (3) Asynchronous: For the ease of setup, the papers should allow (LLM) workers to complete their microtasks independently, without the need of synchronized discussions. The instructor pre-selected six papers meeting these criteria (the first six in Table 1), and students could propose additional papers for approval (Task Paraphrase in Table 1). Up to four students could sign up to replicate the same pipeline in a first-come-first-serve manner.

LLM version Students are required to use text-davinci-003\(^3\) for their final implementations and testing, the most capable model that uses the autocompletion interface at the time of assignment design. However, they were encouraged to initially experiment and fine-tune their prompts using more cost-effective models (e.g., text-ada-001).

Replication assessment We evaluate the replicated chains on two dimensions:

1. Replication correctness: We measure the success of replication using the peer grading results. A replication is considered successful if the average peer score for Correct Replication is greater than three.

2. Chain effectiveness: We evaluate whether the replicated chains are more effective than the baselines using the students’ own assessment. If students indicate that their replicated chains outperform the baselines on the majority of their tested inputs (recall that they were required to test at least three inputs), then the pipeline is deemed effective.

Since multiple students replicated the same pipelines, it is also interesting to compare replicas for the same pipeline to reveal key factors for successful replication. We look into students’ replication strategies, and report the number of (3) Unique replicas. Specifically, we manually grouped the students’ LLM chains based on the microtasks involved, deeming two chains identical if they include steps that essentially serve the same intended functionality, even if there are wording differences in the LLM prompts.

4 Results and Reflection

4.1 Replication Overview: Partial Success

As shown in Table 1, all the pipelines are replicable with LLMs. For each pipeline, there is at least one correct replication and an effective one. To denote the successes, we show one actual input-output sequence generated using students’ LLM chain replications that they found preferable. These results re-iterate that LLMs can now accomplish a subset of tasks that were previously considered possible only for humans (Bubeck et al., 2023).

Several students documented multiple pipelines they experimented with, echoing the prior observation that pipelines/chains enable rapid prototyping (Wu et al., 2022a). Some of their explorations focused on single steps (e.g., P7 in Find-Fix-Verify choosing between different wordings among “fragment”, “clauses”, “substrings” etc.), while some other students experimented with globally redesign-
ing certain pipeline connections (e.g., P11 in Iterative Process varied how the prior results should be passed onto the next step). Interestingly, by examining students’ final submissions and their own reflections, it becomes evident that (students believe) certain pipelines require adjustments (e.g., Microtasking, and Find-Fix-Verify), while others can be replicated more literally (e.g., Map-Reduce).

That said, most pipelines did not achieve 100% success or effectiveness. Students largely attributed the replication failure to prompting challenges — “translating the pipeline into a LLM required a lot of work (in terms of figuring out the correct prompt + the pre-processing that was required in order to move from one step in the pipeline to the next) compared to when it was implemented with crowdsource workers” (P14, Find-Fix-Verify). However, we believe there are more nuanced reasons underlying these prompting difficulties. In the following sections, we delve into several qualitative observations that have emerged from the replication practice and reflect on their implications (an overview of observations and opportunities are in Table 2b).

### 4.2 Replication Variance: Impacted by students’ perceptions on LLM capabilities

One interesting aspect that emerges from the results is that some pipelines have more replication variance than others (i.e., different students’ replications to the same pipeline differ from each other significantly). For instance, while both papers provided sufficient details for replication, the three participants replicating Iterative Process arrived at similar chains. The only difference was created by P11 who introduced another step for choosing top previous results to show subsequent workers, i.e., original steps were not changed.

However, the three students replicating Find-Fix-Verify implemented quite different versions (Figure 2): P14 mostly followed Bernstein et al. (2010)’s descriptions (e.g., having a voting mechanism in the Verify step for reducing human errors), but extended the Find step to include a lot more types of writing issues. They also designed their prompts “using data structures that are easily understandable by a computer versus natural language”, because the LLM “has a background with computer code”. P7, on the other hand, only dedicated the Find step to locate phrases that can be shortened, and instead implemented the Verify step to fix grammatical errors that arose during the preceding shortening steps. They explained that they consciously reshaped the design because they believed that “LLMs do not have these issues [of the high variance of human efforts and errors].” However, this belief is arguably inaccurate. Just
Figure 2: The original pipeline and the LLM replications for (A) Iterative Process (Little et al., 2010) and (B) Find-Fix-Verify (Bernstein et al., 2010). While only P11 diverged from the original Iterative Process pipeline by adding a condition about how previous results should be ranked and used in subsequent steps, students replicating Find-Fix-Verify all had different Verify steps (marked in red box). The chains are slightly simplified for readability.

Reflection: Establish task-specific and pipeline-specific best practices for using LLMs. The different implementations of crowdsourcing pipelines with LLMs showcase the varying assumptions students hold regarding the performance of these models in completing certain tasks and the amount of instruction needed. Indeed, with the rapid advancement of LLMs and prompting techniques, it is challenging to keep up with LLMs’ capabilities and limitations, as well as how they can be applied to specific use cases. Instead of trying to form general mental models about constantly evolving LLMs, it may be more beneficial for practitioners to...
...cally adjust their understanding of LLM usefulness based on the context of their specific use cases. To achieve this, practitioners can adopt a mindset that views LLMs as “Jack of all trades, master of none/few” (Kocoñ et al., 2023), and employ a systematic approach to specifying instructions, gradually moving from sparse to granular. Practitioners can start by establishing a baseline using a general and under-specified prompt, with the assumption that LLMs possess sufficient world knowledge to interpret ambiguous requests. In the context of Find-Fix-Verify, it might be sufficient to implement the Find step with a high-level command like “output any errors in the text,” without specifying error types. Then, if dedicated prompt testing (Ribeiro, 2023) reveals instances where the general prompt falls short, practitioners can adjust their prompts to incorporate more specific instructions, such as textual instructions on corner cases, or employ prompt ensembling techniques (Pitis et al., 2023).

On the other hand, it appears that students have overlooked the fact that LLM performs probabilistic generation during their replication practice, despite being aware of this through their own experiences and course instructions. It is intriguing to observe how the non-deterministic nature of LLM tends to be disregarded, particularly when used in a chaining context. This oversight may stem from a trade-off between creating prototype chain structures and fine-tuning individual prompts for each sub-task (Wu et al., 2022a): LLM’s non-determinism is typically presented using model confidence or the probability associated with the generated output, which may become a secondary consideration when students can only pass on a single output to the next sub-task. To address this, introducing LLM non-determinism as “noises exposed through voting of multiple LLM workers” could allow for integration of disagreement mitigation techniques like adaptive decision-making on the number of votes/annotations needed (Lin et al., 2014; Nie et al., 2020).

4.3 Replication Effectiveness: Affected by LLM vs. Human Strengths

So, what are the actual strengths and weaknesses of LLMs, and how do they affect replicated LLM chains? We delve into students’ reflections on the implementation effectiveness and their ideas for improvements. We find that, not too surprisingly, crowdsourcing pipelines proven effective might require some redesigning to accommodate the unique capabilities of LLMs, which still differ from humans’. This observation aligns with discussions in prior work (Wu et al., 2022b; Webson et al., 2023); however, with the comprehensive exploration of the task space through replications, two significant patterns now become more apparent:

**LLMs need explicit information foraging** Multiple crowdsourcing pipelines require implicit information selection and integration. For example, in Map-Reduce, workers performing the Reduce step had to remove unnecessary information to make the final paragraph coherent. Despite the necessity, few pipelines involve such explicit sub-tasks for selection. This might be because humans are capable of implicit information filtering, re-ranking, and selection (Piroli and Card, 1999; Sperber and Wilson, 1986; Marchionini, 1995). When it is clear that certain pieces are low-quality, out-of-place, or redundant, humans would proactively remove the unnecessary parts so as to retain a reasonable cognitive load. In contrast, LLMs struggle with information foraging, and tend to constantly accumulate context and produce outputs with mixed quality. Students observed these deficiencies at three levels and proposed possible changes:

- **Fail to mitigate low-quality intermediate results.**
  For example, when writing paragraphs with Price-Divide-Solve, P4 found that even conflicting information from different sub-tasks would get integrated into the final writeup, resulting in incoherence (e.g., claiming the university mascot to be both a Scot and an owl). Several students stressed the need for intermediate quality control, for “reducing the unpredictability of the model.” (P13, Iterative Process).

- **Fail to selectively perform a subset of sub-tasks.**
  This is most visible in HumorTool, which, in its original design, required workers to self-select and sort a subset of sub-tasks (eight in total) into an effective flow. Among the four students replicating it, only P17 noticed that the sub-tasks have “no clear structure in the execution order of these micro tasks”, and successfully implemented a chain of four sub-tasks. Other students agreed that eight sub-tasks aggregated too much information, and P18 later reflected that “the steps should not be in such a strict order”.

- **Fail to balance multiple requirements in one sub-task.**
  Excessive requirements in one LLM prompt can also cause conflicts. In the aforemen-
tioned *HumorTool* case, integrating results from too many sub-tasks may lead to certain options dominating others, e.g., the LLM can “focus on turning the joke into being sarcastic, which can take away the humor from the joke” (P5). Similarly, P14 (in *Find-Fix-Verify*) implemented their *Find Step* (Figure 2) to simultaneous searching for multiple issues, which led the LLM to prioritize spelling errors and miss wordiness problems. Overall, explicitly stating the top criteria seem important for LLMs.

**LLMs are more sensitive to comparison-based than humans.** As prior work has observed, LLMs are still sensitive to minor paraphrases (e.g., P7 in *Find-Fix-Verify*) prototyped different wordings among “fragment”, “clauses”, “substrings” etc. in their prompt). However, on the flip side, LLMs are quite responsive to comparison-based instructions. We will use *Iterative Process* for illustration. In its original design, Little et al. (2010) reported anchoring bias to be an inherent limitation of the pipeline: “perhaps owing to the fact that crowdworkers will iterate & improve upon existing ideas, the variance is lower.” All three students replicating this pipeline made similar observations but also found that such bias could be mitigated just with straightforward instructions. For example, P11 initially observed that the pipeline “tends to converge on a specific theme”, but was able to redirect the model with a simple prompt: “The following ideas are examples of low quality, please avoid these common pitfalls.” Similarly, P3 was pleasantly surprised by how effective it is to simply “ask for outputs that differ from the initial set” — “I was originally concerned that providing examples would ‘prime’ the model to generate only examples in the same format, but it seems that this is not an issue in practice.” Note that such simple instructions are unlikely to work for crowdworkers who are trapped by their personal biases (Wu et al., 2021).

This sensitivity to adjectives such as “different” and “diverse” warrants further exploration. One peer grader highlighted this by suggesting, “If we’re allowed to make suggestions, we could ask for titles that are happier, more obtuse, and funnier, which goes beyond traditional crowdsourcing methods.” This suggestion aligns with existing prompting techniques like Self-Refine (Madaan et al., 2023), where LLMs critique their own outputs to generate improved versions focusing on specific dimensions.

**Reflection: Examine effects of instruction tuning, and train humans for complementarity.** While differences between humans and LLMs are expected, it is interesting how some of these disparities arise from the goal of training LLMs to mimic human behavior. For example, methods like Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) use human preferences to enhance LLMs’ ability to follow instructions. This might have simultaneously enabled LLMs to iterate on content based on abstract comparison commands *more effectively than humans*, who often get trapped by cognitive bias or struggle with ambiguous or vague instructions (Gershman et al., 2015). That said, it is unclear whether LLM generations are always better in these cases, as these models are also biased by their training and can have polarized stands (Jiang et al., 2022; Santurkar et al., 2023).

Branching out from this observation, it would be interesting to explore potential “side-effects” of the LLM training schema. Prior work has highlighted the trade-off between few-shot vs. zero-shot capabilities and the need to train LLMs with multifaceted human feedback (Wu et al., 2023). Considering LLMs’ need for explicit information foraging, another worthy line of investigation would be the completeness and clarity of instructions. As most existing instruction tuning datasets prioritize high-quality and precise instructions (Longpre et al., 2023), it remains unclear how LLMs would respond to ill-defined prompts or instructions containing irrelevant information. It might be interesting to examine how LLMs can be trained using a “chain-of-instruction-clarification” approach, similar to the back-and-forth dialogues employed by humans to elicit design requirements. For instance, incorporating a sub-task that involves humans clarifying the top criteria could potentially enhance LLMs’ ability to handle multiple requirements effectively.

The split of strengths also calls for human-LLM complementarity. Instead of humans or LLMs completing all sub-tasks, an effective task delegation among a mixture of different “workers” might be useful. For example, P15 in *HumorTool* noticed the partial effectiveness of their LLM chain: It excelled at “extracting relevant attributes of a news headline and brainstorming associated concepts” but failed at translating them into actual jokes. As such, explicitly training humans to identify and develop skills complementary to LLM strengths could be an
interesting direction to pursue (Bansal et al., 2021; Ma et al., 2023; Liu et al., 2023). Note that this complementarity can occur between humans and a variety of LLMs. For example, P3 in Iterative Process found that while using a weaker model either alone or in a pipeline resulted in poor performance, “when I provided examples from a stronger model as the previous examples [for the weaker model to iterate on], the performance dramatically improved.” This observation reflects that even less-state-of-the-art models can be effective teammates if given the appropriate task — “All models are wrong, but some are useful.” (Box, 1976).

4.4 Replication Challenge: Multi-Modal Regulations vs. Textual Instructions

When reflecting on challenges in LLM replication, four students mentioned the difficulty of creating structured input/output formats. For example, P7 (replicating Find-Fix-Verify) described including a constraint in their prompt: “These segments need to be present in the text.” They stressed its importance in the reflection: “Without this prompt, the returned segments are often sentences dramatically restructured based on the original text, making it difficult to insert them back into the original text after the fix step.” Similarly, P6 in Task Paraphrase said “the major weakness of these prompts was the challenge of extracting structured information out, especially for the pipeline models.”

It is worth considering why human workers, who are as (if not more) “generative” as LLMs, are capable of producing structured inputs and outputs. Essentially, all of the LLM replications of crowdsourcing pipelines are partial — the assignment focuses only on replicating the instructions of the crowdsourcing pipeline, while other components of crowdsourcing are disregarded. Specifically, nearly all crowdsourcing pipelines inherently include constraints introduced by the user interface. For example, in Find-Fix-Verify, the Find step prompts crowdworkers to identify areas for abbreviation through mouse selection on text, guaranteeing that the segment is precisely extracted from the original document. Similarly, He et al. (2015) required annotators to label their questions and answers in a spreadsheet interface with limited answer length and predetermined question options. These ensure that all the answers can be short phrases to predictable questions. Meanwhile, since LLM modules/workers are solely driven by textual instructions, they need additional regulation to compensate for the absence of UI restrictions.

Some students offered textual versions of syntactic constraints, e.g., “a prompting system that allows for much stricter templates (such as the use of a [MASK] token) would make crowdwork-style pipelines much easier.” (P11, Iterative Process). Other ways might also be possible, e.g., transforming generative tasks into multiple-choice tasks so the LLM only outputs a single selection.

Reflection: Alignment in instruction modality, and its role in human simulation. With the emergence of multi-modal foundation models (OpenAI, 2023; Ramesh et al., 2022), it becomes crucial to not only contemplate the alignment between humans and models in terms of instruction following but also to explore the optimal modality of instruction that aligns with human intuition. For example, while LLMs have automated some interactions with visualization, prior work has found that users need mouse events to resolve vague references in their natural language commands (“make this bar blue” (Wang et al., 2022c; Kumar et al., 2017)). Instead of converting such actions into textual instructions, it would be more advantageous to shift towards utilizing visual annotations.

Such challenges also have an impact on the practical applications of LLMs. In the ongoing discussions regarding whether LLMs can faithfully simulate humans, researchers have begun investigating the feasibility of using LLMs as pilot study users for efficiently refining study instructions and designs (Hämäläinen et al., 2023). Indeed, this direction is valuable — Just like in Figure 2, both humans and LLMs need “prompting” to complete tasks. Nevertheless, our findings indicate that such a transition may not be straightforward: On the one hand, since LLMs only respond to textual instructions, an important post-processing step might be required to map LLM instructions into multi-modal constraints for humans. For example, instruction “extract exact sentences” might need to be mapped to an interface design that involves selecting specific phrases, and “paraphrase the main idea” would require disabling copy-pasting from the text to discourage direct repetition and encourage users to provide their own input. On one other hand, as mentioned in Section 4.3, LLMs and humans may respond differently to the same instructions. This discrepancy makes LLMs unreliable even for sim-
ulating human responses to tasks based solely on instructions. We suspect LLMs can be useful for helping study designers reflect on their high-level requirements (e.g., determining what types of human responses to collect), but the literal instruction has to be redesigned. Exploring which parts of the user study design can be prototyped using LLMs seems to be an interesting future direction.

5 Discussion and Conclusion

In this work, we study whether LLMs can be used to replicate crowdsourcing pipelines through a course assignment. We show that the modern models can indeed be used to simulate human annotation in these advanced “human computation algorithms,” but the success and effectiveness of replication varies widely depending on the nature of subtasks. Further, LLMs’ performance and modes of failure can be unintuitive, and they lack the ability to take advantage of multimodal cues that enable human workers to reliably annotate data.

Our qualitative findings indicate two important points: First, examining LLMs within established pipelines or workflows allows for a more straightforward understanding of their strengths and weaknesses, as different pipeline components have different requirements. Second, when utilizing LLMs to simulate human computation, it is advantageous to not only focus on the inherent alignment between human and LLM outputs but also consider aligning additional scaffolds. This involves adapting existing techniques that tackle challenges such as misinterpretation of instructions by humans, noise in human responses, and the need to incorporate multi-modal constraints for humans. Still, due to the setup of the course assignment, the LLM chain qualities varied greatly by students’ efforts and expertise. In addition, given the restricted sample size, quantitative analyses would have yielded limited significance. Future work can look into more systematic investigations on what components of crowdsourcing pipelines could benefit from the use of LLM annotation, and which should continue to be annotated by humans.

From an education perspective, we found having students interact with LLMs actually helped calibrate their confidence in these models — Many students conveyed their frustration when LLMs did not perform as effectively or reliably as they had anticipated. We hope the work can inspire future exploration on allowing students to interact with LLMs and gain awareness of these models’ mistakes, thereby facilitating a constructive learning process and preventing excessive reliance on LLMs. We open-source the assignment design and student responses at https://github.com/tongshuangwu/llm-crowdsourcing-pipeline.

References


Harrison Chase. langchain: Building applications with LLMs through composable.


Guangyuan Jian, Manjee Xu, Song-Chun Zhu, Wenjuan Han, Chi Zhang, and Yixin Zhu. 2022. Evaluating and inducing personality in pre-trained language models.


