Automation of AI R&D: Researcher Perspectives

David Owen
Epoch AI
david@epochai.org

August 27, 2024

Abstract

This report investigates AI's potential to automate AI R&D by interviewing researchers on their work, automation predictions, and evaluation of AI capabilities. Participants identified engineering tasks as more automatable than idea generation, highlighting challenges in reasoning, novelty, and reliability. They predicted near-term automation will focus on coding, ranging from improved assistance to autonomous agents. Most participants predicted that solving existing AI evaluations for engineering tasks would significantly accelerate AI R&D. They also suggested improvements, such as more challenging open-ended tasks and fine-grained assessment of reliability. By clarifying researchers' perspectives on automation, these results could inform AI forecasting and the design of AI R&D evaluations.

1. Executive summary

Background: automating AI R&D might accelerate AI development

The question of when and how AI might automate AI R&D is a crucial topic for AI forecasting. There is a long history of researchers considering this question in the abstract, and describing its importance for how AI will shape the future (Turing, 1951).

Cutting-edge AI models have demonstrated capabilities that may be relevant to AI R&D. For example, software engineering is a large part of AI research. Recent advances have established superhuman performance in coding contest questions (Li et al., 2022), and coding assistants such as Copilot are used at a large scale (Friedman, 2021). Other notable advances include using AI to help develop efficient matrix multiplication algorithms (Fawzi et al., 2022), to optimize data center cooling (Evans and Gao, 2016), and to aid the design of microchips used for training AI models (Liu et al., 2023a). Already, there are examples of AI directly contributing to AI R&D through supervision (Bai et al., 2022) and synthetic data (Zelikman et al., 2022).

These advances underline the transformative potential of AI across many scientific and engineering problems. However, they also raise questions about how to manage these technologies. If AI could meaningfully accelerate AI R&D, then the pace of change might become rapid. For society to understand, govern, and respond to these technologies, it would be beneficial to understand the timelines on which we expect key advances in automation.

Goals

Several research groups have begun developing evaluations and benchmarks to assess AI's capabilities in areas such as software engineering, training run optimization, and machine learning (ML) implementation. This has begun in a piecemeal fashion, identifying tasks that intuitively seem relevant to AI research and creating evaluations based on them. This report brings empirical grounding to evaluation efforts, and to broader questions around automating AI R&D. Concretely, this report has three goals:

1. Characterize AI R&D work tasks, to form a better understanding of where automation may be easier, and where its impacts would be largest.
2. Elicit expert beliefs on the potential of AI to automate parts of their work. Existing surveys have shown widespread disagreement. We hope that detailed interviews may clarify why researchers disagree, and provide testable claims for future work.
3. Collect expert input on proposed AI evaluations for AI R&D tasks. This feedback can inform the design of evaluations, including what tasks to cover, and how they should be implemented.

Methods

To achieve these goals, we conducted semi-structured interviews with eight AI researchers and engineers. We asked them about their day-to-day work, their beliefs about automation of AI R&D, and their thoughts on model evaluations for AI R&D tasks. In addition to being asked about

evaluation design in general, participants were prompted with examples of AI R&D evaluations currently under development, to provide feedback on their scope and implementation. Reflecting recent trends, these evaluations were focused on AI agents: systems that use LLMs as a source of reasoning and incorporate further features such as working memory or tool use. We then analyzed participants' responses for recurring themes, reporting recurring areas of agreement and disagreement.

Findings

Figure 1 provides a visual summary of this work's key findings on the AI R&D workflow, including work tasks, examples, and predictions for automatability. Participants' descriptions of their work tasks fell into six high-level areas: creating hypotheses, designing experiments, running experiments, analyzing results, communication, and studying other work. Participants highlighted several distinctions within these high-level areas, for example between high-level planning of research directions and detailed planning of experiments; or between engineering of prototypes and performance optimization for an established system.

Predictions for automation of AI R&D differed substantially in pace and extent, but all participants focused their responses on software engineering tasks such as implementation and debugging. Other tasks, such as data generation or curation, were highlighted as promising by two participants, but received less dedicated discussion in predictions.

Two participants were extremely optimistic, predicting significant automation of R&D engineering over the next five years. These participants predicted that, in five years, AI agents would autonomously implement code and experiments from natural language descriptions provided by researchers. Their subsequent descriptions of time usage suggested this could automate half or more of their current work. At the more pessimistic end, two participants predicted that AI coding assistants and similar tools would continue to improve, but with a modest effect on AI R&D. One participant described a 20-50% productivity improvement in their work as an extremely optimistic upper bound. The remaining four participants fell broadly between these extremes, predicting that AI assistants would improve, and be helpful, but with little potential to fully automate their tasks.

Participants identified several challenges that AI must overcome to automate AI R&D work: reliability, open-ended planning, long-context reasoning, deep reasoning, and novelty. Due to these challenges, participants unanimously said that automation over the next five years would be focused on implementation tasks, such as coding and debugging, as opposed to hypothesis creation or analyzing results. Several other factors might prevent automation in practice, such as researchers needing to be closely familiar with implementation details, or high compute requirements for running agents, particularly if they run their own ML experiments.

Figure 2 provides a graphical summary of participants' suggestions for AI R&D agent evaluations, including feedback on existing evaluations covering experiment implementation and debugging. Six participants predicted that AI systems that could solve these evaluations would be capable of significantly automating AI R&D. Some participants' descriptions of their work tasks suggested agents capable of solving these evaluations might automate half of current researcher work hours. A caveat to this finding, emphasized by one participant, is that the overall effect of automation could vary greatly across different research tasks, and the overall impact might be much greater (or less) than an estimate of researcher work hours might suggest. Two participants were more skeptical of these evaluations, objecting that R&D relies on more open-ended, challenging tasks.

When asked about evaluation design in general, five participants suggested measuring productivity gains for researchers using AI tools, rather than evaluations of autonomous agents. Researchers' preferences regarding evaluations generally aligned with their predictions about automation in the next five years. The two most automation-optimistic participants unequivocally suggested evaluations for R&D agents. The two participants who were pessimistic about near-term automation, meanwhile, were most emphatic about studying researcher productivity. Other participants saw benefits to both approaches.

Researchers offered several suggestions for improving the evaluations, such as variations on the proposed tasks to increase evaluation robustness. In addition to ideas for evaluating key practicalities, such as reliability, they also offered many examples of tasks to prioritize for future R&D evaluations. A repeated suggestion was to evaluate tasks that are time-consuming but potentially amenable to automation, such as reproducing errors. Two participants also offered further examples of particularly difficult tasks, such as open-ended performance improvement, which might prove to be bottlenecks even if easier tasks could be automated.

Conclusion and next steps

Participants' descriptions of their work, and predictions about automation, clarify how researchers think about automatability in AI R&D. This is in contrast to previous studies of automatability, which tended to use pre-existing rubrics or ratings, and applied these broadly across a large number of occupations and high-level tasks (Owen and Besiroglu, 2023). Our work unearths several points of agreement, even among researchers with differing views about short-term automation. Most researchers predict that AI agents capable of implementing well-defined experiments and debugging errors would significantly accelerate their

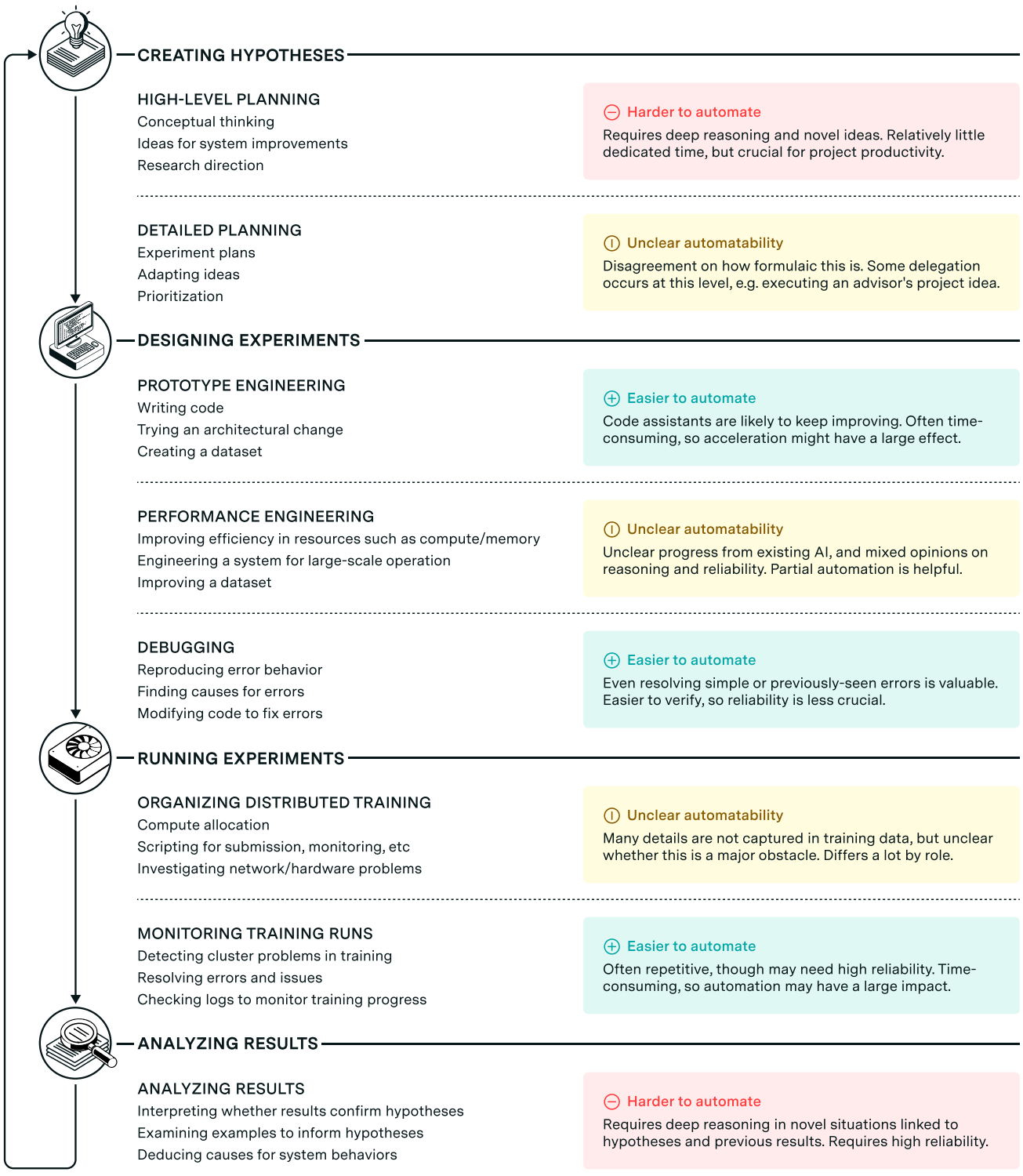

Figure 1. An overview of this work's findings on the AI R&D workflow, including work tasks, examples, and automation predictions.

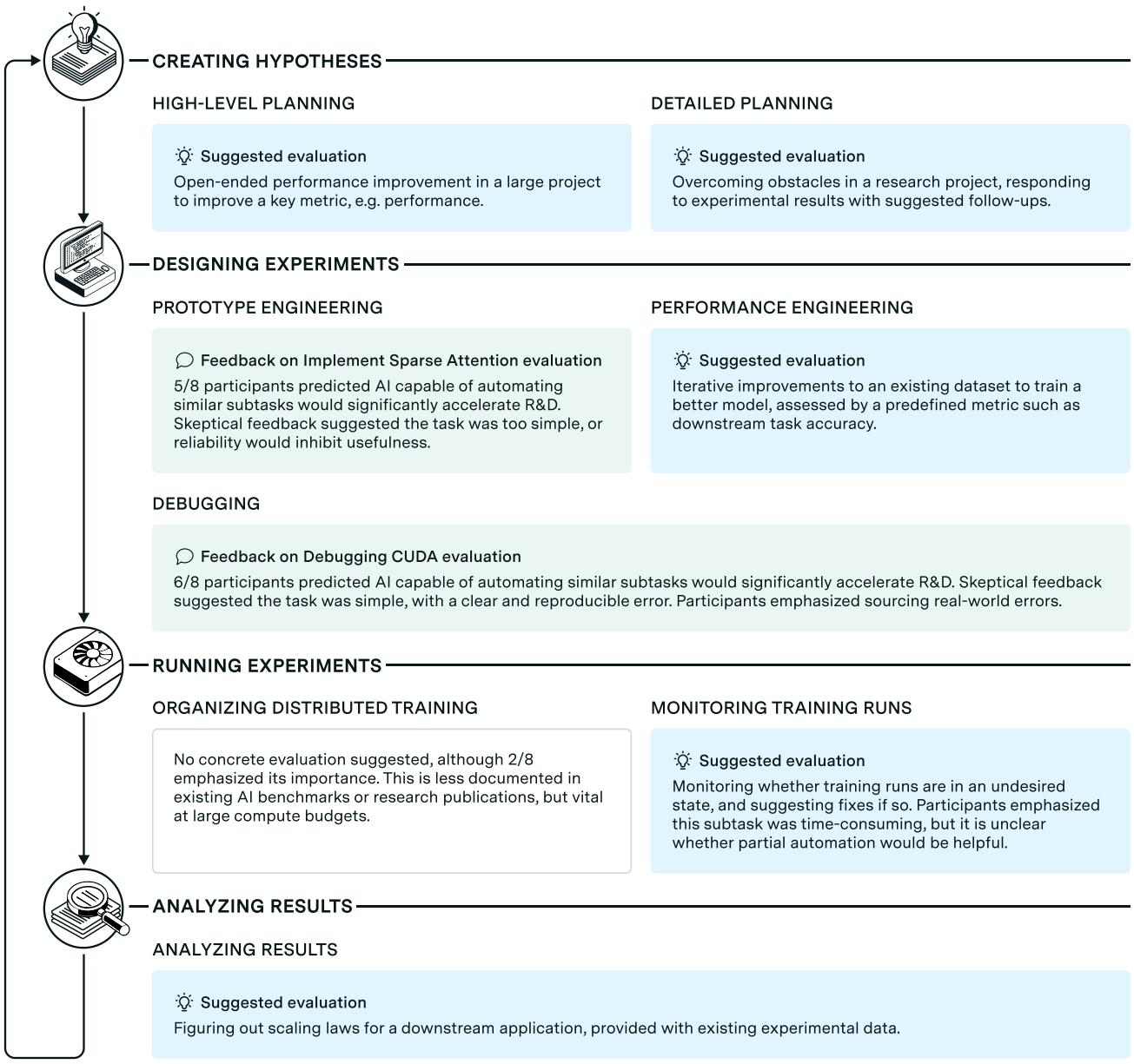

Figure 2. An overview of findings on AI evaluations for R&D, including suggestions and feedback on existing evaluations.

work. Disagreements are primarily about when such agents might become feasible, and whether partial automation will have a significant impact in the meantime.

These findings open several directions for future research. A natural next step would be to measure time spent on R&D tasks. By monitoring activity in detail, and linking this to the bottlenecks researchers identified, it might be possible to produce an empirically-grounded model of automatability. Taking this further, it might even be possible to link such a model to progress in relevant AI evaluations.

In this work we explored whether existing R&D *tasks* might be automated, but it is unclear how this would translate into faster progress at the project level. This depends on many hard-to-predict details, such as compute bottlenecks and potentially diminishing returns to increasing output on currently-important tasks. Several of these crucial details were mentioned by participants in this work. A promising direction could be to study R&D productivity at the project

level; for example, by measuring the productivity effects of existing AI automation in small-scale research projects. Existing research in software engineering suggests current tools have a modest but measurable effect. Quantifying this effect within AI R&D could set a useful baseline to monitor whether subsequent advances drive significant acceleration.

Evaluation design is a closely related problem. To what extent should evaluations focus on full automation, as opposed to assistive AI tools? Our findings paint a mixed picture. Most participants predicted the impact of hypothetical autonomous agents would be significant, but were skeptical that AI would achieve this capability level in the near future. This suggests that agent evaluations might be most useful for detecting an extreme outcome: rapid and substantial automation of AI R&D. Our work offers several suggestions for evaluation design, based on firsthand suggestions from AI researchers. We hope these suggestions may be useful for designing evaluations across a range of tasks and difficulties. Such evaluations, building on existing work by several researchers, promise to become an informative signal for AI's progress in R&D capabilities.

2. Introduction

This report is structured in four sections. In Background and review, we first review relevant literature, existing results, and other background material concerning automation of AI R&D and related tasks. In Methods, we set out details of our methodology for expert interviews and their analysis. In Results, we analyze the interviews and present key results. We split this across the broad areas covered in the interview: Characterizing AI R&D work tasks, Surveying researchers on automation of AI R&D, Designing evaluations for AI R&D automation and suggestions to improve them. Finally, in Discussion and conclusions, we discuss broader implications and next steps.

3. Background and review

We briefly review three strands of pre-existing research: the economics literature on modelling and predicting automation; research and engineering literature on automation of AI R&D tasks; and the AI research literature on evaluations and benchmarking, with a focus on AI R&D software engineering agents.

3.1. Automation in general: economics and prediction

Much of the existing literature on automation comes from economics. There is a small but growing subfield of labor economics studying the automatability of work tasks, and examining the implications of automation for employment, wages and output (Owen and Besiroglu, 2023). This provides a useful starting point for our work in two ways: providing a framework for thinking about automation, and offering some (contrasting) predictions to build upon.

Existing literature makes two fundamental points about automation: automation occurs at the level of tasks rather than occupations, and automation can take different forms with different implications. The task-focused view emphasizes that occupations can undergo transformation in some tasks even as other tasks remain unchanged. The second point, that automation takes different forms, is closely related. Broadly, there are three forms of automation: full automation of tasks, with technology entirely substituting for labor; partial automation, with technology improving workers' productivity and acting as a complement to labor; and deskilling, reducing the skill requirement to perform a task. For our purposes, these points suggest it is important to consider *which tasks* are being automated, *how* they are being automated, and *changes* from rearrangement of tasks as a result.

The second way the automation literature is relevant to this work is predicting automation. Several authors have devised different approaches for this challenging task. They broadly fall into three categories: rating task features, such as broad skills required and their amenability to automation; mapping descriptions of tasks to innovations, for example patents; and surveys to elicit estimates of automation, for example when all of the tasks in an occupation will be automated.

The track record for task feature rating and patent mapping predicting automation is mixed. There is some evidence that, across the entire economy, they can be weakly predictive of changes in hiring and wages, explaining perhaps 1-10% of these in previous waves of automation. Putting aside challenges in interpretation for these findings, such methods cannot answer more detailed questions such as, "When would it be possible for AI to automate half of the researcher-hours involved in developing a frontier LLM today?"

An advantage of surveys is that they specifically cover AI research, and look further ahead to predict substantial automation. Moreover, they question AI researchers, who are by definition most familiar with AI R&D. Surveys show massive variation in researcher timelines: in a recent survey, over half of researchers assigned at least 50% chance of all AI researcher tasks being automatable by 2044.[1] However, individual predictions ranged from years to centuries (Grace et al., 2024). There was more agreement that AI research is among the harder occupations to fully automate. These findings are a useful starting point: we wish to elicit descriptions of *why* people have shorter or longer timelines, tasks they predict being automated, and what the effects will be.

[1] Strictly, this result came from researchers' median timeline for high-level machine intelligence, which by definition included AI research. However, other phrasings led to different results, suggesting the forecasts are not robust.

3.2. Automating AI R&D: research and tools

The AI literature covers automation of AI R&D, not from a perspective of prediction and analysis, but instead with an aim to develop AI models and techniques to perform AI R&D tasks. By AI R&D, here we broadly refer to the work tasks of AI researchers and engineers. We do not focus on related work that could significantly affect AI R&D, but is not directly part of AI researchers' work, such as chip design or creation of new numerical primitives. This narrows the scope of our work, making the question more tractable.

What are the work tasks of AI researchers? From the academic literature, one description focuses on "experimentation, the process of designing and running experiments, analyzing the results, and iterating towards some positive outcome" (Huang et al., 2024). The particular details of these tasks vary substantially, reflecting different areas of AI research. Clearly, there are also tasks that do not fall under this description, such as disseminating findings, studying previous work, or managing team members. Nevertheless, this captures the dynamics of R&D on open problems.

One area with plentiful existing research is the automation of data science tasks, particularly in AutoML (De Bie et al., 2022). AutoML typically refers to full automation of well-defined steps in applying ML methods to new problems, such as model selection and hyperparameter tuning. To a large extent, the success of AutoML reflects pre-AI (or at least pre-neural) automation. There are exceptions: architecture search has benefited from neural reinforcement learning (Zoph and Le, 2016), and there is recent proof-of-concept usage of language models to assist in AutoML applications (Zhang et al., 2023).

Other applications of neural AutoML remain in their infancy, although AI-assisted data curation has shown promising results for generating high-quality datasets for both general-purpose pretraining and capability-specific post-training (Shao et al., 2024). Relatedly, the use of AI-generated synthetic data has shown great promise for improving mode capabilities, and is incorporated in current frontier models (Dubey et al., 2024).

Another important area of existing research is software engineering. Here, we see real world adoption of AI assistance via tools such as Copilot and ChatGPT. Existing tools seem to improve software engineering productivity, perhaps more so for lower-skilled users (Ziegler et al., 2022). These tools hence fall more into the categories of partial automation and deskilling.

At the cutting edge of research on AI for software engineering, there have been many notable advances in code repair (Xia et al., 2023), code generation (Friedman, 2021; Roziere et al., 2023; Allal et al., 2023; Li et al., 2022), and fixing bugs (Keller and Nowakowski, 2024). There has been striking progress from small, closed tasks towards more open-ended problems. Recent research has increased the autonomous agency of AI systems by using LLMs as a source of reasoning (Yao et al., 2022; Shinn et al., 2024), while incorporating a working memory (Auto-GPT, 2023; Yang et al., 2024a; Huang et al., 2024), tools such as web browsers and code execution (OpenAI, 2023a; Anthropic, 2024b), and planning modules to break down tasks into subtasks (Yao et al., 2022; Wei et al., 2022; Yao et al., 2024). Today's agents can sometimes solve longer software engineering tasks end-to-end, for example resolving issues from real world software repositories (Jimenez et al., 2023).

Finally, and most directly relevant to this work, there is research directly focused on AI R&D tasks. In addition to assistive tools, such as ML systems to help with literature search (Wang et al., 2023) or coding (Friedman, 2021), there is also agent-focused research, where models are developed to *perform* the R&D tasks directly (Liu et al., 2023b; Huang et al., 2024).

3.3. Evaluations and benchmarking

Evaluation of AI capabilities via benchmarks is a familiar practice in machine learning. Generally, *benchmarks* refer to a set of reusable problems with a straightforward procedure to score results and hence measure performance. *Evaluation* is the broader process of estimating model performance, incorporating benchmarking but also experimentation and interpretation to assess a model's capabilities (METR, 2024). Recent evaluations of AI agents increasingly use a relatively small number of tasks, involving significant computational resources and human intervention to execute (METR, 2024; Anthropic, 2024a).

Several authors have begun developing agent evaluation tasks that are relevant to AI R&D (Department for Science and Technology, 2023; METR, 2024), and frontier AI labs have discussed incorporating AI R&D capabilities in their policies for model development (Anthropic, 2023; OpenAI, 2023b; Shevlane et al., 2023). To the extent evaluations already exist, they broadly follow the areas described in Section 3.2: specific AI R&D tasks and software engineering. We particularly focus on evaluations for AI agents, although many of the software engineering benchmarks are applicable to assistant tools.

MLAgentBench stands out as an early benchmark to measure AI agents' capability to autonomously perform R&D experimentation (Huang et al., 2024). This covers 13 ML tasks "from diverse domains ranging in difficulty and recency" - for example image classification on CIFAR-10, Kaggle challenges, and open ML research problems. Somewhat similarly, MLBench tests the ability of agents to replicate tasks taken from the READMEs of several ML projects (Liu et al., 2023b). Finally, the METR task suite

Table 1. AI R&D evaluation tasks used in the interviews.

R&D work task	Specific evaluation
Implementing well-defined experiments, and reporting the results.	Replace attention with sparse attention in a provided codebase and set of pretrained weights. Finetune and evaluate performance.
Debugging errors.	Debug a codebase with a CUDA stream concurrency error.

contains a small number of challenging ML research evaluation tasks, such as building AI for a board game or replicating an ML paper (METR, 2024).

There are many benchmarks for software engineering. At the time of writing, one of the most challenging benchmarks in the literature is SWE-Bench, which harvested GitHub issues to find real-world software engineering problems (Jimenez et al., 2023). Previous programming benchmarks typically considered smaller self-contained problems (Austin et al., 2021; Shinn et al., 2024; Cobbe et al., 2021; Chen et al., 2021). As might be expected, agents perform better on SWE-Bench compared to prompting models in a straightforward "single-turn" setting (Yang et al., 2024a;b). The METR task suite and OpenAI evaluation suite also offer several evaluations focused on software engineering, of varying difficulty and scope (METR, 2024; OpenAI, 2024).

4. Methods

4.1. Interview structure

We conducted one-hour interviews with each subject. These interviews had a two-part structure, with further detail on interview structure available in Appendix B:

1. Characterizing AI R&D work and predictions around automation. Researchers were asked to describe their day-to-day work, challenges for automation, and their predictions around AI automation of AI R&D tasks in the future.
2. Evaluation design for automation of AI R&D. Researchers were presented with example evaluation tasks and asked to assess their significance and relevance to automating AI R&D, as well as thoughts on evaluation design. Researchers were also asked more general questions about how they would design evaluations to detect AI automating AI R&D.

Example evaluation tasks for the second part are described in Table 1. These were selected from ongoing work on AI R&D agent evaluations. These evaluation tasks were prioritized for two reasons. First, they are designed to represent real tasks in ML research, even if in a simplified form. Many existing evaluations consider problems such as Kaggle competitions, which are further removed from the real work of AI researchers. Second, these tasks are calibrated for difficulty. They are too challenging for present-day systems to solve, but less challenging than tasks such as "prepare a solution for a recent unsolved ML research challenge".

4.2. Subject recruitment

We invited ML researchers or research engineers who have either published at leading conferences such as NeurIPS or ICML, or who had similar experience. We focused on researchers with experience of either (i) creating architectural or algorithmic innovations used to improve training or inference; (ii) devising post-training enhancements to improve model capabilities; (iii) contributing to the development of large models.

We recruited participants via direct invitation. Eligible participants were identified based on (i) recent ML publications in relevant academic venues; (ii) pre-existing professional connections; or (iii) referral from either of the first two categories. We ultimately interviewed eight participants - this provides a sense of how researcher opinions differ, and even which opinions are more prevalent.

Four participants had experience developing LLMs in frontier industry labs, or developing leading open-source models. For three of these participants, their involvement was more focused towards model training, for example developing training algorithms and engineering training runs. The remaining participant with large model development experience was focused on evaluation of already-trained models. The four participants without experience developing large models were researchers in post-training enhancements, architectural/algorithmic innovations, and interpretability.

Three participants were, or recently had been, employed in large industry AI labs. Two participants were employed in open-source efforts. Two participants were graduate students in academic groups. The remaining participant was employed at a start-up after completing a PhD. Six participants were primarily individual contributors, albeit with in-depth technical expertise and some managerial respon-

sibilities. Two participants described their jobs as focused more on management than individual contribution.

4.3. Analysis of interviews

Thematic analysis was used to identify repeated themes and elements across different respondents. We first identified themes in responses to individual questions. This led to an ongoing process of coding responses by participant. Subsequent passes over the data incorporated responses to follow-up questions, identifying further common themes. Ultimately this led to high-level summaries supported by the data, for example, "Overall, six participants mostly expected AI to provide assistance rather than full automation of any significant tasks.".

Where useful, we include supporting quotations from participants. Quotations are edited for clarity. This involves removing fillers such as "um" and "like", repetition, modifiers such as "very", and making other small changes. More significant changes, such as omission of clauses or sentences, are marked with [square brackets].

5. Results

Results are organised under three sections. First, in Section 5.1, we document participants' detailed descriptions of the everyday work of an AI researcher. Second, in Section 5.2, we elicit researcher predictions on automation of their work tasks, and investigate factors that make R&D tasks challenging for AI. Finally, in Section 5.3, we collect researchers' feedback and thoughts on evaluation design, with a particular focus on the predefined evaluation tasks set out in Section 4.

5.1. Characterizing AI R&D work tasks

The first part of the interview focused on researchers' descriptions of their work tasks: the activities in which they spend their time, the importance of these activities for their work, and the way in which they are organised. The aim is to build a better understanding of where automation would have significant impacts, rooted in detailed descriptions of AI R&D work.

R&D IS A FEEDBACK LOOP OF HYPOTHESIS, EXPERIMENTS AND RESULTS

Figure 5.1 characterizes the AI R&D workflow, building on the description of (Huang et al., 2024). For every high-level task in the workflow, most or all participants described corresponding tasks and examples. Further details are provided in Appendix A. These high-level tasks are not strictly sequential, and researchers move back and forth between them. Moreover, this workflow is not limited to any particular area of research, such as developing an architectural change. The same structure holds for dataset research, for example: a researcher must establish a plan, and criteria to know whether they have succeeded, then implement this through engineering to create or modify datasets, then run experiments and analyze the results.

> Participant 1 (P1): "Research goes in cycles where I will spend a bunch of time trying to figure out what I think the important problem is, what I want to work on, and where I think I have some headroom to make some progress. Then I will go and start to think about what it actually is that I want to do to make progress on that area, and that itself looks like a slightly different thing. And then there's sort of raw execution work..."

A key point is that this workflow can be recursive. While implementing an experiment (*prototype engineering*) to study an idea for a new architecture (*creating hypotheses*), a researcher might encounter a bug (*debugging*) or an unexpected result (*analyzing results*). This would trigger a recursive process of creating hypotheses for the cause, and devising experiments to investigate further. Sometimes this can be tightly coupled to the original outer-level loop, for example because the unexpected result reveals important information about the higher-level problem.

There were several distinctions within participants' descriptions of hypothesis generation, design of experiments, and running experiments. In *hypothesis generation*, there is a distinction between high-level planning and detailed planning of experiments. High-level planning was described in various ways, but focuses on project ideation, project management, and intuition about research directions. Detailed planning involves more concrete detail about what to do next, translating high-level ideas and hypotheses into experiment ideas. This distinction overlapped with the differing contributions of senior advisors and their juniors. The fact that this work can be factored separately in existing R&D may be relevant for automation - although, as we discuss later, participants were divided on this and similar questions.

Designing experiments is an even more concrete level of detail: taking plans, then translating them into a functioning implementation that can be used for informative experiments. Within this task, participants described several different examples, which we categorised as prototype engineering, performance engineering, and debugging.

Prototype engineering involves developing the first functioning version of an experiment (P8: "you do something from scratch but as simple as possible [...] you observe something funny with some big model and you don't quite understand it. So you try to boil it down to some really simple example, and go to a Jupyter notebook"). Meanwhile,

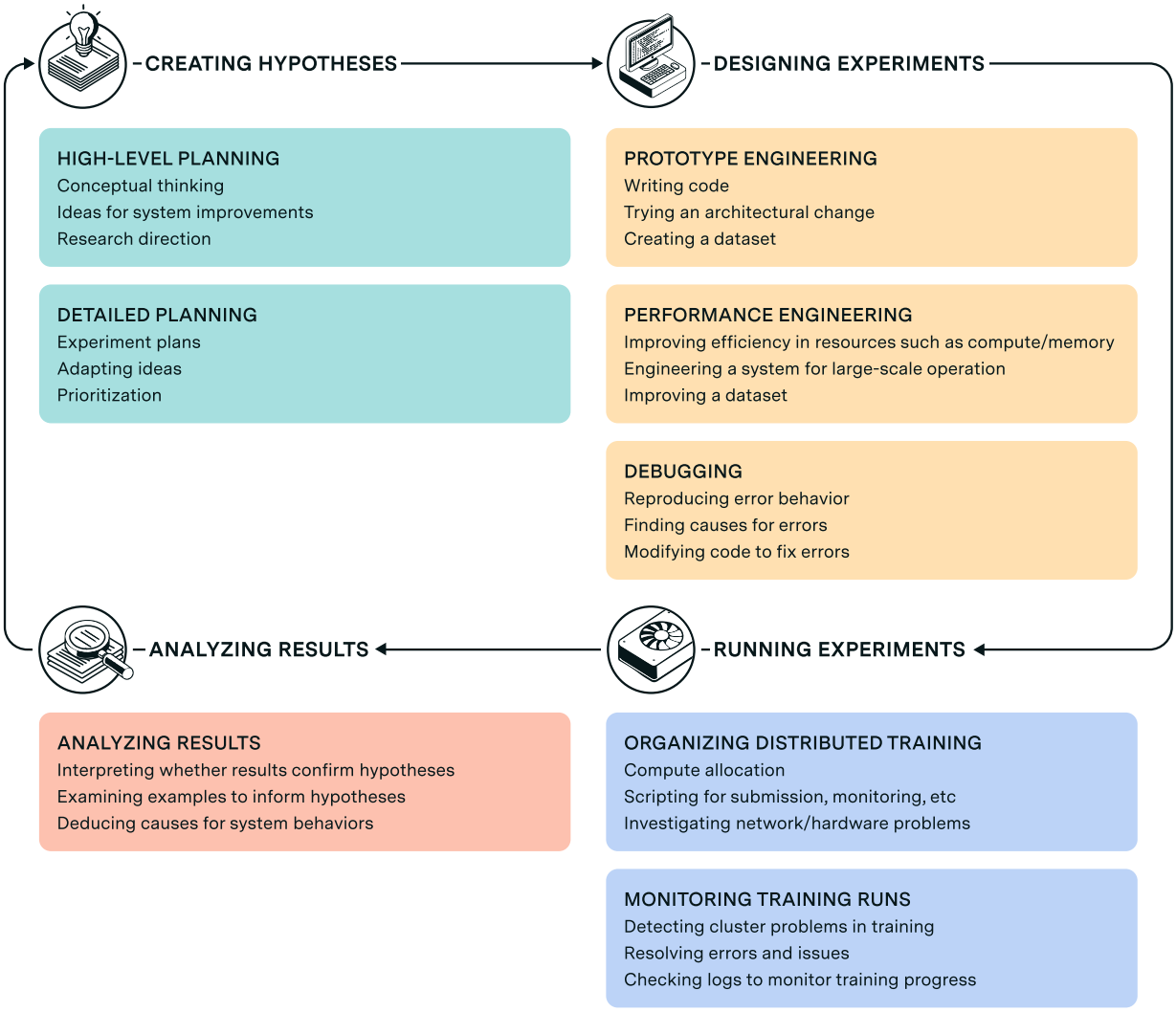

Figure 3. The AI R&D workflow based on participants' descriptions, expanding on the pre-existing description of (Huang et al., 2024). The four high-level workflow tasks decompose into several different tasks, with several concrete examples for each. This figure does not include additional high-level tasks such as studying other work or communication, which are discussed in Appendix A.

performance engineering involves re-engineering existing implementations to improve some fairly well-defined and measurable property, such as speed. Prototype engineering is more likely to involve substantial new additions to a codebase, whereas performance engineering tends to involve modifying an existing mature codebase.

Debugging covers the common tasks of investigating and fixing undesired behaviour. Debugging is tightly coupled with the other activities; for example, implementing ideas in code for the first time often leads to bugs. Nevertheless, at their extremes, the ideas seem separable. Participants talk about switching back and forth between implementing the idea, encountering issues, and then debugging them (P3: "for something like UL2, which is fairly complex, I probably need five iterations of debugging").

Experiment design is followed by running experiments. Running experiments decomposes into subtly different tasks: organising the allocation of resources and practicalities, and monitoring ongoing experiments. These differences were present in only two participants' descriptions, but in both

cases they were significant. For example, P4 described painstakingly maintaining a spreadsheet organising training jobs, and consulting documentation for the scheduler to submit them. This heterogeneity seems related to participants' work and teams - participants who work on small-scale experiments barely mentioned running experiments as its own set of tasks, whereas participants directly involved with large-scale experiments described a significant challenge.

In addition to the predefined R&D tasks, communication and studying other work were repeated themes. Six participants discussed the importance of communication in their work. For the most part, this was about informal collaboration within a project. Formal communication, such as research papers or documentation, was often described more as an afterthought. Finally, seven participants explicitly mentioned studying other work. Typically this was literature review in the early stages of a project, described as distinct from hypothesis creation.

Hypothesis creation is vital but quick; engineering and debugging are time-consuming

Six participants emphasized that planning and hypothesis creation are extremely important within a project. It is unclear how to specify the time these take, due to overlap with designing experiments, studying other work, etc. Nevertheless, participants' descriptions suggest these are quick, to the extent that they have their own dedicated time.

> P3: "The decisions you make in the beginning are really important and since there's nothing to start with you really have to think from first principles [...] and also finding the topic to work on because that's such high leverage. If you choose a topic that's uninteresting or not important then you know the rest of the project... it essentially goes that direction."

All participants described the importance of engineering and debugging, and how time-consuming these can be. This was a core part of participants' work in most cases, although more senior participants described spending less time on this as they shifted towards managing other researchers.

> P6: "Most positions in AI nowadays are a combination of research and engineering whether or not that's actually your title, and insofar as you're not actually doing engineering, it's almost always directly managing people who are doing the engineering."

5.2. Surveying researchers on automation of AI R&D

This section examines researchers' predictions about automation of AI R&D work tasks. The aim is to examine predictions in detail, investigating where and why researchers disagree, as well as their descriptions of challenges for automation, and how these relate to work tasks.

Predictions differ on automation extent, and partial automation's usefulness

There was significant disagreement on the potential of AI to accelerate AI R&D in the next five years. Table 2 categorizes participants' outlooks, and provides examples of their predictions and reasoning. Broadly, six participants were optimistic about automating AI R&D. Four were only somewhat optimistic, describing significant improvements in AI assistance but little full automation, while two were extremely optimistic, describing rapid progress from assistance to full automation on significantly many R&D tasks. Two were more pessimistic, believing AI assistance is (and will remain) mostly helpful for more routine tasks.

Six participants mostly expected AI to provide assistance rather than full automation of any significant tasks. More common was a description of having a better Copilot, or tools for inspecting code for common mistakes. Participants who did describe delegating substantial tasks to AI mostly envisioned that these would be around implementation of experiments, with researcher labor directed towards planning and analysis. We discuss their reasons further in the next section, What makes automation of R&D hard?

What makes automation of R&D hard?

Participants identified several challenges for automation: reliability, open-ended planning, long-context reasoning, deep reasoning, and novelty. Many of these are self-explanatory, with the arguable exception of deep reasoning. The examples linking these limitations to R&D work, set out in Table 3, are illustrative. For example, deep reasoning examples involved anticipating the implications of design or implementation decisions across interconnected areas, creating and reasoning over abstractions, and other necessary reasoning to enable research success. Several of these challenges (open-ended planning, deep reasoning, novelty) are linked to creating hypotheses and analyzing results. All participants suggested that implementation tasks such as coding and debugging will be easier for AI in comparison.

All four participants who discussed reliability raised the issue that for unreliable AI solutions, it would be necessary to inspect an unreliable AI's work. If this inspection was time-consuming, it might undermine any benefit from automation. P2 offered the example that, in their experience, current AI code generation causes errors that take time to debug. P2 further claimed that whereas a human software engineer can explain parts of their code, AI explanations are weaker, and less coupled to an understanding of original intentions. This shows that even in partial automation,

Table 2. Participants' outlook for automation of AI R&D over the coming five years. Most participants stressed that their predictions had low confidence, and they were offering their view on the likeliest scenario. Circles show how many participants held each view.

Outlook	Quote
Pessimistic ●● AI assistance might be useful for easier software engineering, but not much for R&D.	P1: "I don't see effort being applied as much in terms of how do we get AI to lift the simplest truth out of all of the dirty mess that the data provides [...] next token prediction seems a far cry away from that behavior and I think that's the behavior that would really unlock both research abilities for AI, for self-improvement, as well as mathematical abilities, and top-notch software engineering, and all sorts of other things."
Somewhat optimistic ●●●● AI assistance will keep improving, and this will be helpful for AI R&D, but few tasks will be fully automated.	P4: "Coding is definitely something that people are working on right now, and they are making significant progress. This is something that is a very big part of people's lives, and it's taking up all the time from people. So if coding is accelerated... I think it's mainly about acceleration of the code."
Extremely optimistic ●● AI assistance progresses to full automation of significantly many tasks.	P7: "Humans will just be talking in natural language to this huge model, and the model would both manage the code base and run the experiments, and basically it might also manage the upcoming training run."

reliability can be an important challenge.

Novelty is clearly a challenge for current AI models: tasks that are far outside a model's training data are hard. Two participants described this as a bottleneck for automating R&D in particular. Another subtlety, raised by the same participants, was that tasks with low legibility also may not be covered in training data. P5 emphasized that much of their work would not be stated in a typical research paper; rather, it was dealing with issues in large compute clusters. P1 offered a similar example of reverse-engineering the hardware design of accelerators in order to write better-performing CUDA code.

Two participants raised a practical concern about fully automating implementation tasks: the need for researchers to be familiar with implementation details. Even if an agent could successfully implement experiments, a researcher may need to scrutinize the details in order to build up their understanding of the problem. This step may be necessary to successfully understand their results and generate further hypotheses. P8 was particularly skeptical that R&D work could benefit from automating implementation tasks.

> P8: "If you're doing something that's never been done before, then writing it out yourself and seeing all the decisions you have to make, and keeping track of those to try to debug it later when something goes wrong, is part of that process and part of what's necessary in order to do good research there."

Finally, three participants suggested compute bottlenecks could be another practical obstacle for automation, even when AI capabilities are ready in principle. Other practicalities, such as integration into the code environment, were mentioned, but participants stated these would likely be solvable. However, they were less confident that automation would be compute-efficient. These participants suggested labs might allocate spare compute towards experiments and scaling, if automation of R&D tasks was compute-intensive.

5.3. Designing evaluations for AI R&D automation

This section focuses on design of evaluations. The aim is to collect expert feedback on proposed evaluation tasks for AI R&D agents. This involves assessing the significance for automation of these work tasks, and the implementation details necessary for evaluations to accurately represent them. This section also discusses researchers' suggestions for other evaluation tasks.

Implementation and debugging are promising work tasks for evaluations

If evaluation tasks only covered a small part of real AI R&D work, solving them would have little impact on AI R&D. To assess evaluation scope, we asked researchers to

Table 3. Limitations of AI that currently stand in the way of automation. Circles show the number of participants who raised a limitation, and empty circles show participants who predicted this challenge would substantially improve over the next five years.

Challenge	Description
Reliability ●●○○	P5: "You ask it to implement this method from this paper and test it out and report back, but then if it fails by producing code that doesn't compile, and then returns that it failed this task, are you willing to assume that architecture doesn't work, or do you think that it's just the agent made a mistake?"
Open-ended planning ●●○○	P6, offering an open-ended example: "Our model is only at 75% on this score. That's not acceptable, we need to get up to 80%. I'm going to leave it to you [the agent] to figure out how to do that. [Tasks like this] will be much more common in the world and also harder to automate."
Long-context reasoning ●●○	P2: "We're seeing a lot of marketing about, 'Hey our context window is three million, ten million tokens, just throw your entire code repository in and now we can implement functions, you can effectively do updates on an entire repository.' But my counter-argument to that would be that I think that the ability to process a larger context length is orthogonal to your ability to actually use that context."
Deep reasoning ●●	P1, describing the design of CUDA code: "What is the abstraction that both captures most of what I want to do with deep learning while also respecting what the hardware is looking for? [...] That's something I think we're pretty far away from, in both context ingestion but also just reasoning. There's a bunch of reasoning that you need to do to make that work and in GPT-4 the reasoning seems a little bit too shallow."
Novelty ●●	P6, describing how AI R&D involves novel methods that wouldn't work at smaller scales: "Most things an AI has done, some human has done [...] for the frontier tasks, figuring out how to make things run at extremely large scales or coming up with new training methods for AI, they haven't ever worked on small models, but now work."

describe the significance for their work if AI agents could perform the work tasks in the leftmost column of Table 4. These were selected as fairly well-defined proposals from in-progress R&D evaluations, as discussed in Section 4.1. Six participants predicted that either or both of these tasks, if automated, would have a substantial effect on their work, such as automating half or more of their current work-hours.

Five participants predicted that an agent able to autonomously implement and run well-defined experiments would significantly accelerate AI R&D. There was wide disagreement over the extent of implied automation, ranging from little-to-none ("hours or days in a multi-month project", broadly the position of three participants) to a great extent ("60% of my time would be automated if there was a model that is doing that", broadly the position of five participants). Pessimism mostly stemmed from concerns that R&D relies on more open-ended, challenging implementation tasks.

The five participants who predicted that an experiment-implementing agent would accelerate R&D expressed positive views that the specific evaluation was a good representation of this work task. In fact, the evaluation corresponded closely to one participant's work. P4 described spending most of a month implementing different variants of sparse attention from the literature and testing them out. P7 distinguished between different senses in which well-defined could be meant: it could mean that the relevant examples were in the agent's training data, or it could mean that the agent was provided with relevant documentation in a similar way to a human researcher. The former sense would be less useful, as relevant examples are fairly unlikely to be in training data for novel R&D work.

The debugging of errors evaluation requires an agent to autonomously investigate and fix bugs such as a CUDA stream concurrency error. Participants were prompted for feedback on this evaluation, unless they had already discussed debugging of their own volition earlier in the interview. Six participants suggested this capability would meaningfully accelerate AI R&D, although participants were reluctant to quantify the significance in terms of time spent. An exception was P3, who explained how "if everything after the time it doesn't work counts as debugging then it's probably like 70% [of time spent coding]".

Again, a crucial detail was whether a hypothetical agent needed to cover the most challenging, open-ended bugs to be

Table 4. Work tasks from evaluations, and the significance of automation they might imply if an agent could autonomously perform them. Circles show the number of participants with optimistic or pessimistic predictions for automation if AI could solve an evaluation.

	Work task	
	Well-defined experiments	**Debugging errors**
Evaluation	Replace attention with sparse attention in a provided model codebase and set of pretrained weights. Finetune and evaluate performance.	Debug a provided ML codebase with a CUDA stream concurrency error.
How much AI R&D automation (optimistic)	P4: "60% of my time would be automated if there was a model that is doing that" ●●●●●	P3: "If everything after the time it doesn't work counts as debugging then it's probably like 70% [of time spent coding]." ●●●●●●
Why this might have significant impact	P4: "Some of my project was specifically on that, basically implementing different kinds of attention [...] I spent at least a month doing different variants of sparse attention."	P5: "Sometimes [errors] are very difficult to reproduce or don't always appear, and those are definitely very tricky to find [...] but I don't think those need to be automatically solved for it to be helpful."
How much AI R&D automation (pessimistic)	P8: "hours or days in a multi-month project" ●●●	P8: "If none of your code's throwing an error anywhere [...] that is much harder to debug but I think would be infinitely more useful, in that case." ●●
Why this might not have significant impact	P8: "That particular bit of implementation is not that complicated, not that complex."	P8: "There's no explicit CUDA bug, but I'm getting the wrong results and I can see mathematically they're wrong results, and that's very hard to debug."

useful. Six participants stressed that bug difficulty can vary greatly. For this reason, participants were more equivocal about how well the specific evaluation represented this part of their work. Three participants emphasized that debugging explicit errors might be easier than debugging unexpected or undesired behaviours without an explicit error message. However, three participants predicted that accelerating even relatively easy debugging tasks could be significant.

> P5: "I think it would still be helpful in the sense of either easily suggesting next steps for actionable experiments to try, or helping and generating potential patches for easier ones. You can just follow the suggestion, after observing it, and then going through and not being slowed down by simpler errors. I think that would still be helpful."

Intriguingly, the two participants with pessimistic predictions for forthcoming automation also predicted that solving the specified evaluations might not entail substantial acceleration of R&D. This suggests their pessimism about automation is compatible with having similar predictions for AI capabilities improvements as other participants. They simply disagree that such capabilities would be useful to automate AI R&D.

Agent evaluations might not track real-world automation

The evaluations examined in these interviews were based on AI agents autonomously performing R&D tasks. This is a common feature across much existing work on evaluation suites for automation of AI R&D, as discussed in Section 3.3. However, existing AI applications for engineering tasks are

more focused on partial automation using assistive tools: coding assistance, better search for technical information, etc. This difference was noted by several participants, three of whom were particularly concerned that agent evaluations might not track real-world automation progress.

Concern over full versus partial automation matches participants' earlier predictions, discussed in Section 5.2. Although six participants predicted that solving the agent evaluations would entail significant automation of R&D, most participants previously predicted that R&D automation in the next five years would be from improving assistive tools rather than autonomous AI agents.

When asked how they would design evaluations, five participants suggested measuring productivity gains for researchers using AI tools, rather than evaluations of autonomous agents. Researchers' suggestions for evaluations aligned with their predictions about forthcoming automation. Extremely automation-optimistic participants suggested evaluations for R&D agents. Participants who were somewhat optimistic described benefits for both approaches. However, there was a notable exception. P1, despite skepticism of agents accelerating R&D in the near future, explained that this might nevertheless be a crux for sudden and substantial acceleration of AI R&D.

> P1: "The question really is, in my view, in what situations does AI become a complement to labor versus a substitute for labor? [If] I'm going to co-pilot a little bit while you're doing this it can be at most, in my view, a 20 to 50% improvement, and that's pretty optimistic."

Evaluations should cover many variations and examine reliability in depth

Participants offered several ideas to expand or improve on evaluations, shown in Table 5. A repeated suggestion from four participants was to cover different variations and difficulties within high-level tasks such as "implementing a well-defined experiment". This could prevent overindexing on a single evaluation that happens to closely match examples in training data. This could also be more informative about the limitations of AI models (P7: "Current models are quite brittle, so I think they will have high variance and unpredictable swings in what they can count on").

> P3: "You've got to make sure it covers very different scenarios where the features are very different, the codebases are of different lengths and sizes, but other than that I think it's a good benchmark [...] that would be fairly easy to test."

Similar suggestions for debugging involved evaluating many different bugs in different codebases. Three participants suggested sourcing bugs at a variety of difficulties from real world examples, although one noted this poses a risk of data leakage.

> P7: "Maybe I would look on StackOverflow for people asking, 'I'm getting this obscure bug, can someone help me?', and then I would read the solution and if it looks like wow, that's a crazy bug... I would try to invent something similar to that one. But yeah, it's not perfect because then maybe the model can remember this specific example really well. It could just apply the same kind of reasoning to the task in the eval set."

Another repeated suggestion, from two participants, was to focus on reliability, for example by grading best case versus average case performance. This matches participants' earlier discussion of reliability as an obstacle to full automation of tasks. P5 suggested assessing performance at a fine-grained level within evaluations, which might give a better sense of early progress and challenging subtasks.

> P5: "Split 'implement new attention mechanism' into maybe 15 different steps. Can it write the code, and then can it run the proper experiments, or can it select hyper parameters? Instead of all of those things being different aspects that could be accomplished but not accomplished sequentially."

Priorities for new evaluations: time-consuming tasks and difficult tasks

Participants offered several further ideas for evaluation tasks. A repeated suggestion across three participants was to prioritize tasks that occupy a lot of researchers' labor but may be particularly automatable (P1: "What you'd really need is for there to be some subset of things that are very labor-intensive [...] like babysitting training runs"). P8 suggested a related idea: tracking researcher time usage and looking for time-consuming problems, such as extremely challenging bugs. P8 emphasized that this might be a necessary step to ensure evaluations focus on the correct tasks. P5 highlighted that none of the example evaluation tasks specifically examined dataset creation and curation, despite this occupying significant research effort.

Two participants suggested evaluating tasks that seem particularly challenging to automate, which might become bottlenecks to overall automation of R&D. These were often less clearly defined than the labor-intensive tasks, and are shown in Table 6. They are generally close to the AI limitations identified in Section 5.2, for example exploring open-ended problems, deep reasoning, and research intuition.

Table 5. Potential weaknesses in evaluation tasks and corresponding improvements, where participants provided them.

Evaluation	Weakness	Suggested improvement
Sparse attention	Reasoning too easy	Range of problem difficulties
	Needs high reliability or verification	Robustness across different problems, codebases, runs
	Might not work on novel codebases/ideas	Test on novel codebases
	Might only work for easily-isolated changes	Similar tasks that imply wider changes, e.g. new parallelism strategy
	Insufficiently open-ended	Separate evaluation of more open-ended R&D problems
	Depends on small experiments remaining useful for future R&D	-
Debugging CUDA	Explicit errors versus incorrect results	Test more ambiguous errors
	Errors unrepresentative	Try to source from real world
	Training set contamination	Source from outside training
	Large scale may be important	-

6. Discussion and conclusion

Disagreements on automation are substantial, but this disagreement provides useful information. Researchers have vastly differing intuitions about *when* hard problems will be solved, but mostly not what those hard problems are. Within the AI R&D workflow, researchers are fairly unified that implementation tasks are more amenable to automation than planning, reasoning and interpreting results.

Characterizing AI R&D work tasks in terms of bottlenecks offers insight into what tasks may be automated sooner, and where to target evaluations to detect incipient automation. Several of these bottlenecks, such as deep reasoning or novelty, correspond to bottlenecks identified in the literature on automation (Brynjolfsson et al., 2018). Here, they re-emerge spontaneously from researchers' description of real AI R&D work tasks. Other bottlenecks are more closely linked to the specifics of AI R&D and LLM agents, such as making effective use of long context.

One promising direction for future work could be investigating these bottlenecks for fine-grained R&D tasks, trying to identify which seem hardest to automate and why. Previous work has already shown the potential of a similar methodology for high-level rating of automatability across the economy (Brynjolfsson et al., 2018). Developing this further would require careful definitions for these properties, but raises the possibility of developing expert-grounded forecasts for automatability.

Most participants predict AI accelerating R&D via increasingly powerful assistive tools. When asked how they would design evaluations for automation of AI R&D, these participants favor measuring researchers' productivity increase when using AI tools (Ziegler et al., 2022). However, when they considered evaluations for autonomous agents, most participants predicted solving these would entail significant automation of their work. Hence, agent-based evaluations might be useful to specifically detect *rapid* acceleration of R&D via automation. Plausibly, it might be best to adopt a

Table 6. Tasks that participants suggested to include in a suite of AI R&D evaluations, or otherwise highlighted within these categories.

Task	Description
Labor-intensive tasks	
Monitoring and adjusting training runs	P1: "Something that's very labor-intensive like babysitting training runs."
Reproducing errors, bugs and behaviours	P5: "Sometimes [errors] are very difficult to reproduce or don't always appear and those are definitely very tricky to find and require a lot of effort."
Dataset creation and curation	P5: "Tweaking the data mixture and seeing how things perform and then evaluating."
Communication and prioritization for a contributor or manager	P7: "I think a lot of things are also about communication between different people [...] you don't see as many projects being like, 'Let's make a product manager or an automated manager.'"
Harder-to-automate tasks	
Investigating open-ended problems	P6: "Open-ended performance improvement, where nothing's wrong but you've got to change a thing of your choosing to make the performance go up."
Experiments at a small scale to predict large scale behaviours	P7: "Figuring out scaling laws for a particular application."
Challenging reasoning to figure out next steps	P6: "Okay, I got some error back, here is the error message. I'm like bouncing back and forth there, and doing zero shot evaluations of the form, 'What do you do now?' You ask the model and then read off what happens."
Overcoming obstacles	P7: "And then the second aspect that you might want to test is how good it is at handling problems or overcoming obstacles. So it could be that it would have to install a library and encounter problems, would have to fix bugs."

portfolio of evaluations, with productivity measurements examining assistive tools, and capability evaluations focused on autonomous agents.

Participants offered several suggestions for how to design evaluations, and AI R&D tasks to prioritize in an evaluation suite. Generally, these were focused on difficult, open-ended tasks with longer horizons. Present day models are unable to solve such tasks end-to-end, which might be addressed with grading of individual stages within the evaluation. Such an approach, although difficult to design, could offer a detailed view of progress. A valuable next step would be developing evaluation tasks in further detail, including grading of stages. These evaluations could be submitted for detailed feedback from researchers, similar to in this work.

A key limitation in this work is its qualitative nature. Researchers assessed how much of their own work might be automated by agents that could solve certain evaluations, and sometimes this was even expressed as a percentage of their time or impact on a project. Inevitably, these estimates were approximate and speculative. A promising direction for further work would be to quantitatively measure time usage through screen recording or similar means. Such a

project could also be valuable for detecting changes in R&D work, providing further evidence about its automation and the impacts thereof.

Nevertheless, detailed reports from AI researchers are a useful starting point for examining where and why they disagree. This work is among the first documenting how AI researchers think about automation of their work, and what they think of evaluations for R&D agents. By collecting researchers' views on this important topic, we hope to improve AI forecasting and the design of AI R&D evaluations.

Acknowledgements This work was funded by the UK AI Safety Institute to build the evidence base on AI's contribution to AI research and development.

References

Loubna Ben Allal, Raymond Li, Denis Kocetkov, Chenghao Mou, Christopher Akiki, Carlos Munoz Ferrandis, Niklas Muennighoff, Mayank Mishra, Alex Gu, Manan Dey, et al. Santacoder: don't reach for the stars! *arXiv preprint arXiv:2301.03988*, 2023.

Anthropic. Anthropic Responsible Scaling Policy, Version 1.0, 2023. URL https://anthropic.com/responsible-scaling-policy.

Anthropic. The Claude 3 Model Family: Opus, Sonnet, Haiku, 2024a. URL https://www.anthropic.com/claude-3-model-card.

Anthropic. Claude can now use tools, 2024b. URL https://www.anthropic.com/news/tool-use-ga.

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.

Auto-GPT. Auto-GPT Documentation, 2023. URL https://docs.agpt.co/.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Erik Brynjolfsson, Tom Mitchell, and Daniel Rock. What can machines learn and what does it mean for occupations and the economy? In *AEA papers and proceedings*, volume 108, pages 43–47. American Economic Association 2014 Broadway, Suite 305, Nashville, TN 37203, 2018.

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tijl De Bie, Luc De Raedt, José Hernández-Orallo, Holger H Hoos, Padhraic Smyth, and Christopher KI Williams. Automating data science. *Communications of the ACM*, 65(3):76–87, 2022.

Innovation Department for Science and Technology. Introducing the ai safety institute, 2023. URL https://www.gov.uk/government/publications/ai-safety-institute-overview/introducing-the-ai-safety-institute.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

Richard Evans and Jim Gao. DeepMind AI Reduces Google Data Centre Cooling Bill by 40%, 2016. URL https://deepmind.google/discover/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-by-40/.

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Nat Friedman. Introducing GitHub Copilot: your AI pair programmer, 2021. URL https://github.blog/2021-06-29-introducing-github-copilot-ai-pair-programmer.

Katja Grace, Harlan Stewart, Julia Fabienne Sandkühler, Stephen Thomas, Ben Weinstein-Raun, and Jan Brauner. Thousands of AI authors on the future of AI. *arXiv preprint arXiv:2401.02843*, 2024.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating Language Agents on Machine Learning Experimentation. In *Forty-first International Conference on Machine Learning*, 2024.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? *arXiv preprint arXiv:2310.06770*, 2023.

Jan Keller and Jan Nowakowski. AI-powered patching: the future of automated vulnerability fixes. Technical report, Google Research, 2024.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with AlphaCode, 2022.

Mingjie Liu, Teodor-Dumitru Ene, Robert Kirby, Chris Cheng, Nathaniel Pinckney, Rongjian Liang, Jonah Alben, Himyanshu Anand, Sanmitra Banerjee, Ismet Bayraktaroglu, et al. ChipNeMo: Domain-Adapted LLMs for Chip Design. *arXiv preprint arXiv:2311.00176*, 2023a.

Yuliang Liu, Xiangru Tang, Zefan Cai, Junjie Lu, Yichi Zhang, Yanjun Shao, Zexuan Deng, Helan Hu, Zengxian Yang, Kaikai An, et al. ML-Bench: Large language models leverage open-source libraries for machine learning tasks. *arXiv preprint arXiv:2311.09835*, 2023b.

METR. Autonomy Evaluation Resources, 2024. URL https://metr.org/blog/2024-03-13-autonomy-evaluation-resources/.

OpenAI. Code Interpreter - OpenAI API, 2023a. URL https://platform.openai.com/docs/assistants/tools/code-interpreter.

OpenAI. OpenAI Preparedness Framework, 2023b. URL https://cdn.openai.com/openai-preparedness-framework-beta.pdf.

OpenAI. OpenAI Evaluation Suite, 2024. URL https://github.com/openai/evals.

David Owen and Tamay Besiroglu. Challenges in Predicting AI Automation, 2023. URL https://epochai.org/blog/challenges-in-predicting-ai-automation.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code Llama: Open Foundation Models for Code. *arXiv preprint arXiv:2308.12950*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 2024.

Toby Shevlane, Sebastian Farquhar, Ben Garfinkel, Mary Phuong, Jess Whittlestone, Jade Leung, Daniel Kokotajlo, Nahema Marchal, Markus Anderljung, Noam Kolt, et al. Model evaluation for extreme risks. *arXiv preprint arXiv:2305.15324*, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Sara Turing. *Alan M. Turing: Centenary Edition*. Cambridge University Press, 1951. Alan Turing, in 1951, predicted "[...] it seems probable that once the machine thinking method had started [...] they would be able to converse with each other to sharpen their wits".

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. Scimon: Scientific inspiration machines optimized for novelty. *arXiv preprint arXiv:2305.14259*, 2023.

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.

Chunqiu Steven Xia, Yuxiang Wei, and Lingming Zhang. Automated program repair in the era of large pre-trained language models. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 1482–1494. IEEE, 2023.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. *arXiv preprint arXiv:2405.15793*, 2024a.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. Intercode: Standardizing and benchmarking interactive coding with execution feedback. *Advances in Neural Information Processing Systems*, 36, 2024b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *Advances in Neural Information Processing Systems*, 36, 2024.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping Reasoning with Reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488, 2022.

Shujian Zhang, Chengyue Gong, Lemeng Wu, Xingchao Liu, and Mingyuan Zhou. AutoML-GPT: automatic machine learning with GPT. *arXiv preprint arXiv:2305.02499*, 2023.

Albert Ziegler, Eirini Kalliamvakou, X Alice Li, Andrew Rice, Devon Rifkin, Shawn Simister, Ganesh Sittampalam, and Edward Aftandilian. Productivity assessment of neural code completion. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pages 21–29, 2022.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

A. Quotes about work subtasks

Table 7. Main R&D workflow tasks, subtasks, examples, and descriptions from participants.

Workflow task	Subtask and examples	Example description
Creating hypotheses	High-level planning Conceptual thinking, ideas, research direction	P1: "Research goes in cycles [...] trying to figure out what I think the important problem is, what I actually want to work on, and where I think I have some headroom to make progress."
	Detailed planning Experiment plans, adapting ideas, prioritization	P4: "The process of formulating the hypotheses is something that is preceding me writing down the actual experiments I'm going to be running [...] For example, you can say this specific way of doing stuff is going to improve or change this parameter of my system by this much [...] you establish an experiment that basically has two distinct possible outcomes that you are going to distinguish between your hypotheses."
Designing experiments	Prototype engineering Writing code, updating the code, trying a new architectural change, creating a dataset	P3: "If I want to create something new [...] start with a really simple baseline and then gradually add complexity"
	Performance engineering Compute, efficiency, scaling, modifying a dataset	P8: "You write a first version and you try to make sure it's correct. And then you time it and see what's slow about it. Try to iterate on the algorithm like that. Yeah, so that's very bad... that type of implementation is a lot more involved than just quick experimentation."
	Debugging Reproducing error behaviour, finding causes for errors, modifying code to fix errors	P5: "Okay, we know we're failing to load this checkpoint from an external library but there must be some problem and it's not entirely clear what the source of that problem is. First, maybe check on a single node or with no parallelism. Check whether the model successfully trains in this case compared to some ground truth or some expected behavior. And then slowly scale up or introduce new variables, maybe train a model architecture we know works. So this seems to work if we put it through the pipeline, what about if we add in this thing?"
Running experiments	Organizing distributed training Compute allocation, scripting, network issues	P4: "I have a big spreadsheet of all experiments that are planned and the timetable of when they are supposed to be running and just filling out that stuff was taking up some part of my time"
	Monitoring training runs Detecting cluster problems in training, resolving errors and issues, checking logs to monitor training progress	P5: "prepping a bunch of different experiments and then launching them and making sure that they don't ever die or fail due to the cluster health or errors midway through"
Analyzing results	Analyzing results Interpreting whether results confirm hypotheses, examining examples to inform hypotheses, deducing causes for system behaviours	P2: "looking through the data of the performance on evaluation to find holes [...] it's performing at 50% on this subtask within this task and a majority of the errors can be accredited to this behaviour"

Table 8. Additional workflow tasks, subtasks, examples, and descriptions from participants.

Workflow task	Subtask and examples	Example description
Communications	Formal writing Reports, papers, documentation	P3: "The easiest part for me is writing out the paper in the end because usually the pieces all come into place and I just have to spend time on on Overleaf writing them all down."
	Informal communication Talking, messaging	P4: "I need to catch up on the communication a lot more because there are some questions or follow-ups about ongoing projects that are blockers [...] some papers are not reproducible, and we need some sources of data, or we need to brainstorm new ideas, or stuff like that.."
Studying other work	Studying Reading papers, studying codebases, attending talks	P3: "if I want to create something new, I would start with a short literature review, check that it's it's really something new and I'm not reinventing the wheel"

B. Interview structure

Prerequisites (5 minutes)

- Introduction and preamble.
- Context, structure, and terminology.

Characterizing AI R&D work (10-20 minutes)

- Walk through a typical day of your work, and the tasks you work on in AI R&D.
- Describe key tasks in more detail.

Predictions about automation (10-20 minutes)

- In your work, are some tasks particularly amenable or not to AI automation in the next five years?
- Why? What does this automation look like? What would be the impacts?

Predefined evaluations (20-40 minutes)

- Presentation of work task and evaluation.
- How much of your time could be saved if AI could automate tasks like this? What concrete examples do you imagine being automated?
- How well does this evaluation represent the task? Would solving this evaluation imply meaningful automation of the real-world task?
- What are the biggest limitations of this evaluation? How could it be improved?

Evaluations in general (5-20 minutes)

- Are there other tasks that you would prioritize for inclusion in a suite of evaluations?
- What do you think about agent-based evaluations of automating tasks? If you were designing a project to detect AI accelerating AI R&D, would you do it differently?