November 2025

Working Paper

Self-building benchmarks: Using AI-generated exams to understand LLM work capabilities

Maria del Rio-Chanona and Johanna Einsiedler

This working paper is available online at: https://www.brookings.edu/center/center-on-regulation-and-markets/

Disclosure

The Brookings Institution is committed to quality, independence, and impact. We are supported by a diverse array of funders. In line with our values and policies, each Brookings publication represents the sole views of its author(s).

Self-Building Benchmarks: Using AI generated exams to understand LLM work capabilities

R. Maria del Rio-Chanona[1], Johanna Einsiedler[2]

[1]University College London [2]Copenhagen Center for Social Data Science, University of Copenhagen, Copenhagen, Denmark
November 19, 2025

Abstract

Benchmarks are essential for understanding Large Language Models (LLM) capabilities yet are costly to develop and update for real world work tasks. Here we use an Agentic AI approach, where LLMs themselves automatically generate and evaluate practical exams for tasks across Finance & Business Operations, Management, and Computer & Mathematics occupations. To develop these exams, we distinguish between materials needed (e.g. text, data, images) and tools required to solve them (e.g. function calling, web search). Focusing on text-only tasks requiring no tool use, we find only 7% (149 tasks) of these occupations are testable. We deploy 13 models including GPT, Claude, and Gemini variants to complete these synthetic exams. Even on basic tasks, current LLMs struggle: leading models achieve median scores of 65-79%, with performance particularly weak on data manipulation and financial calculations. However, models show rapid improvement—those released in 2024 averaged 40.5% while 2025 models reached 66%, a 26 percentage point gain in one year. While considerable work remains in validation and extending to tool use to expand beyond the current text-only testable tasks, our results suggest that LLM-generated benchmarks may offer a cost-effective, scalable, and updatable approach for measuring AI workplace capabilities, extending the "LLM-as-a-judge" paradigm to occupational task assessment.

Keywords: Large Language Models; Future of Work; Artificial Intelligence, Agentic AI; LLM-as-a-judge; JEL codes: J24, J23, C63, O33

1 Introduction

Generative artificial intelligence (AI) is reshaping the job market, creating demand for experienced specialists in developing AI technology while potentially reducing the need for workers performing other technical, translation, and administrative tasks or novice work (Lichtinger and Hosseini Maasoum 2025; Teutloff et al. 2025). Government reports predict these shifts will accelerate, making it vital to pinpoint which skills will matter most across sectors from office administration to finance and information technology (Department for Education (DfE) London 2024). Understanding current and potential future capabilities of Large Language Models (LLMs) is becoming increasingly important, both for firms defining industrial strategy and aiming to enhance productivity (Aghion et al. 2025), and for policymakers who want to determine which skills will be most relevant to help workers adapt. This was prompted a growing interest in evaluating the performance if LLMs in real work tasks.

Both economics and computer science studies have investigated LLM work capabilities. Economists have developed exposure measures, which systematically estimate whether LLMs can perform work tasks across occupations. Early work within this area relied on expert judgment (Frey and Osborne 2017) or crowd-sourcing (Felten, Raj, and Seamans 2021); more recent contributions combine O*NET task descriptions with human or LLM annotations to gauge direct and indirect exposure (Eloundou et al. 2024). Computer scientists have taken a benchmark approach, where they develop evaluations to test AI capabilities on specific tasks. Some of these benchmarks are based on exams commonly taken by humans (e.g. the USA Math Olympiad; Petrov et al. 2025), others are based on developed environments such as WorkBench and AgentBench that simulate office settings to evaluate multi-step assignments (Styles et al. 2024; Xu et al. 2024).

While these studies have made considerable progress, they still face significant hurdles. Exposure-based measures can systematically link tasks to occupations, but their conclusions remain speculative and lack performance evidence. Empirical evaluations offer valuable snapshots of model capabilities, yet quickly become obsolete as technology improves. More recent benchmarks based on simulated work environments demand substantial manual effort, often cover narrow and potentially cherry-picked task sets, and typically lack systematic selection criteria.

To move beyond exposure estimates and theoretical or manually intensive benchmarks, we propose a framework grounded in three principles. First, capabilities are unobservable constructs that must be inferred from performance. In human ability testing, skills are measured through structured tests; in economic complexity, countries' productive capabilities are inferred from their exports (Neffke et al. 2024). We apply this logic to LLMs: capabilities should be revealed through systematic testing. Second, we focus on economically relevant capabilities that produce measurable outcomes rather than abstract knowledge. Third, we require benchmarks that can be updated as technology advances without incurring the cost and delay of manual reconstruction.

Our methodology follows from these principles. We draw on the O*NET taxonomy, which captures economically valuable tasks for which workers receive compensation. We systematically select tasks across occupations rather than choosing arbitrarily. We construct practical tests that require concrete task completion—preparing budgets, generating tax returns—rather than testing abstract knowledge. We then extend the LLM-as-a-judge paradigm (Zheng et al. 2023; Golchin et al. 2024) by implementing an AI agent system that generates benchmarks, administers tests, and evaluates submissions automatically.

Our findings suggest that only a small fraction of tasks (approximately 3–10% per occupation group) can be evaluated in a fully automated and dynamic manner without allowing multimedia input or external tools. Nevertheless, even this restricted set of tasks serves as a measure for tracking improvements in LLM capabilities. Models show consistent gains over time, with state-of-the-art systems achieving median scores above 75%, while smaller or older models remain below 40%. These trends indicate that our method may be useful for forecasting AI capabilities and assessing their potential impact on labor markets. By enabling scalable, domain-relevant evaluations that can be updated without the cost and delay of manual content development, our proof-of-concept contributes to the economic literature on technological forecasting and the computer science literature on AI benchmarking through a systematic framework for evaluating AI performance on realistic job tasks.

The remainder of the paper is structured as follows. We begin by reviewing related literature across fields. Next, we describe our data and methodology for systematically selecting tasks, generating exams and grading them. We then present our results, including task filtering, model performance scores across occupations and types of tasks, and an analysis of capabilities improvements over time. The final section discussed the main implications and promising further work.

2 Related literature

Our work build upon different strands of literature in economics and computer science.

In economics, a substantial body of literature has examined the potential impact of automation on specific job tasks. Frey and Osborne (2017) used expert opinions on occupations exposure to automation and estimated that nearly half of the U.S. workforce faces high automation risk. Building on this approach, Webb (2019) applied natural language processing to show that more complex, high-skill tasks often score higher on AI exposure. Felten, Raj, and Seamans (2023) further expanded these measures by mapping AI application domains to human abilities, offering a multifaceted index of occupational vulnerability. More recently, Eloundou et al. (2024) have enlisted LLMs themselves to rate exposure, concluding that while only 1.8% of US occupations could have over half of their tasks directly affected by LLMs, this share increases to almost 50% of jobs when accounting for likely future development of complementary software. While recent exposure measures agree in that high-wage occupations are most exposed to AI, the extent to which this means substitution or complementarity is unclear (Rio-Chanona et al. 2025). Moreover, these measures rely heavily on expert judgment and remain difficult to validate empirically (Frank, Ahn, and Moro 2025).

At the same time, computer scientists have been struggling to develop measurements that keep up with the rapid improvement of LLMs. Early models were evaluated on next-word prediction accuracy, but this metric quickly lost discriminative power as top models approached optimal performance on standard text prediction tasks. General-purpose language understanding benchmarks such as GLUE (A. Wang, Singh, et al. 2018) and SuperGLUE (A. Wang, Pruksachatkun, et al. 2019)–which assessed models on tasks like sentiment analysis, question answering, and textual entailment–similarly reached ceiling effects as leading models approached or surpassed human performance. This saturation of standard metrics has led to the development of more specialized benchmarks designed to measure a model's capabilities across specific knowledge domains including law (Guha et al. 2023), medicine (Liu et al. 2024), history (Hauser et al. 2024), and mathematics (Hendrycks et al. 2021). More recently, benchmarks of real workplace capabilities have also started to emerge.

Workplace task benchmarks include WorkBech (Styles et al. 2024), a sandboxed office environment where even the best-performing model (GPT-4) completed only 43% of tasks correctly. SWE-Lancer is a benchmark connected to real freelance software engineering tasks, with the best frontier models achieving an earn-rate of 40% (Miserendino et al. 2025). TheAgentCompany (Xu et al. 2024) simulated an enterprise ecosystem where the strongest agent achieved just a 30% autonomous completion rate. These results indicate that many occupational tasks challenge today's models. However, these benchmarks are not systematically aligend with official taxonomies of tasks such as O*NET[1], making it difficult to link their findings to real-world occupations.

[1]Xu et al. (2024) draws inspiration fro O*NET tasks, but focuses on a handfull of occupations are tasks do not match directly to O*NET tasks

Given the high costs and difficulties of benchmarking, researchers have begun using LLMs themselves as evaluation tools. Zheng et al. (2023) found that GPT-4 as a judge achieves over 80% agreement with expert and crowd preferences on LLM outputs. Seungone Kim et al. (2023) introduced PROMETHEUS, a specialized 13B-parameter model trained on GPT-4 feedback that correlates strongly with human graders. Farchi et al. (2024) automated coding task benchmark construction and evaluation through consistency checks. Beyond single-model judgments, Schoenegger et al. (2024) demonstrated that aggregating evaluations from multiple diverse LLMs–a "wisdom of the silicon crowd"–can match the accuracy of human forecasters on real-world events.

Despite these advances, benchmarking and evaluating LLMs still faces significant hurdles. Chang et al. (2023) argue for establishing evaluation as "an essential discipline to drive the success of LLMs" given the complexity of assessing these systems. Saxon et al. (2024) highlight that benchmarks measuring general capabilities often lack construct validity, questioning whether they actually measure what they claim to measure, and call for a new discipline of "model metrology" to generate evaluation methods that better predict real-world performance. Additionally, Panickssery, Bowman, and Feng (2024) demonstrated that when LLMs are used as judges to evaluate outputs, they tend to favor text written in their own style, suggesting potential evaluation biases that could skew benchmark results. Our work draws on these research strands to outline a systematic, updatable approach for assessing LLM performance on realistic job tasks. We focus on a curated subset of O*NET tasks within business and technical occupations, and use an LLM-powered pipeline to both design evaluations and perform assessments. By comparing state-of-the-art models with earlier versions, we track capability development over time.

3 Exam generation

To infer LLM's capabilities on workplace tasks we construct remote, practical exams generated and evaluated entirely by an LLM, hereby referred to as the exam-generating LLM. These exams are then given to an exam-taking LLM.

We orchestrate the pipeline for generating exams with **LangGraph**[2]—an open-source framework that represents a series of LLM calls as a directed graph whose nodes can branch, loop back on failure conditions, and pass structured messages to one another. Throughout this pipeline a *single* model instance (e.g. Claude 3.7 Sonnet) performs every step: exam drafting, sanity checks, and grading. Figure 1 shows an overview of the pipeline which we discuss in detail below.

3.1 Tools, materials and feasibility

The first step to generate the exam was to specify which tools and materials were required, whereby *tools* we referred to software or applications, in particular spreadsheet software, code execution environments, image viewer, image generator, text editor, presentation software, online search engine, video conferencing, chat interfaces, email client, internet access and other specialized software that the candidate needs to use to complete the test. As *materials* we denote files such as text, data, images, audio files, video files, virtual lab environments and other specialized files that form part of the test content. For each work task we asked Claude 3-7 Sonnet to list the materials and tools that are *absolutely necessary* to complete the task. In addition we asked the LLM to assess whether the task can be completed remotely and whether it is feasible to create a remote, practical exam to test an individual's capabilities to complete the task (see Appendix B.1 for the prompts used).

The exam-generating model then constructs exams for specific work tasks through six sequential message exchanges

3.2 Exam design and writing

The exam-generating model then constructs exams for specific work tasks through six consecutive messages exchanges (see Figure 1). At each step, we provide the task name, associated occupation, the previously generated list of required materials and tools, the typical education level for that occupation, the system prompt, and any responses from prior LLM calls. The system prompt assigns the LLM the role of an expert examiner responsible for assessing an occupation's capability to perform a specific task. For example: "You are an excellent examiner of *Tax Preparers'* capabilities. Design a remote, **practical exam** to verify whether a *tax preparer* can *Compute taxes owed or overpaid, using adding machines or personal computers, and complete entries on forms, following tax form instructions and tax tables*". While the system prompt stayed the same throughout the whole exam generation process, the user prompt was used to pass more specific instructions, effectively breaking down the generation process into smaller work packages and passing on all responses from previous steps:

[2] "LangGraph", accessed April 22, 2025, `https://langchain-ai.github.io/langgraph/`

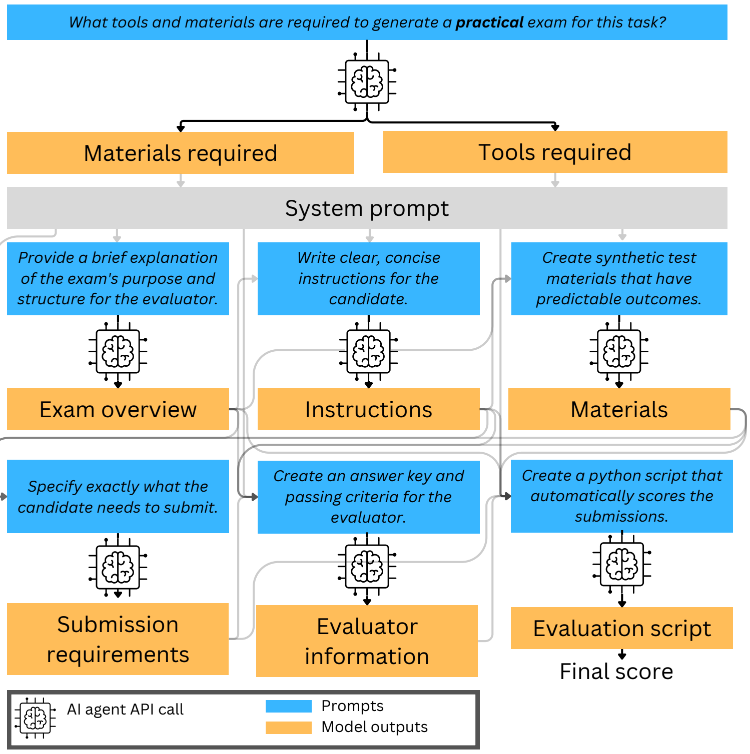


Figure 1: Illustration of the pipeline used for generating the exams.

Step 1: Exam overview. In the first step the LLM was prompted to "provide a brief explanation of the exam's purpose and structure for the evaluator", as well as additional guidance for the exam evaluator by highlighting any strategic elements in the exam design.
Step 2: Instructions. During the second step, the LLM was provided the previously generated exam overview, together with the assignment to generate instructions for the candidate, including the exam's goal, a brief description of the materials that will be provided, the expected format for answer submission as well as the actual test.
Step 3: Materials. In the third step, the LLM was instructed to create synthetic test materials such as e.g. contents of CSV-files or datasets for the candidate as well as accompanying explanations for the evaluator, which were not accessible to the exam-taking model at any stage.
Step 4: Submission requirements. Step four prompts the LLM to generate concrete submission instructions, detailing the required JSON answer format.
Step 5: Evaluator information. In step five, the LLM is tasked with generating an answer key, explanations of the correct answers and grading instructions for the exam evaluator.
Step 6: Evaluation script. The final step asks the LLM to write a *Python* script that can be used to automatically score submissions.

For the full prompts see Appendix B.2.1.

The final exam given to the exam-taking LLM consists of instructions, materials, and submission specifications. The overview and explanation of materials were generated for the exam-generating LLM to be able to write the *Python* script that automatically scored the submissions of the exam-taking LLMs.

3.3 Sanity checks

The exam creation process was accompanied by a few "sanity checks". First, with respect to the material generation step (Step 3), the exam-generating LLM validated whether the answer did indeed contain a section with candidate materials and a separate section with information for the evaluator. If this was not the case, material generation was repeated up to three times. If all attempts failed, the exam was marked as invalid.

After the exam was generated several checks were run, intended to increase the likelihood that the exam (1) did not falsely state that image material was provided, (2) did not reference a fabricated website or news, (3) overall made sense and (4) had an answer key that scored 100% when evaluated using the provided evaluation script. The first three checks were implemented through additional calls to the exam-generator LLM, prompting it to check the above properties. If the answer key scored less than 99%, a new answer key and evaluation script were generated (i.e. steps 5 and 6 were repeated). If the exam failed any of the first three checks or the answer key did not achieve full score after three trials, the exam was marked as invalid. All prompts can be found in Appendix B.3. In addition we also marked exams as invalid if the answer key scored more than 100%.

4 Data

4.1 O*NET

We use O*NET database of the U.S. Department of Labor, Employment and Training Administration as the data source for work tasks and their descriptions as well as to infer the level of education required for the majority of positions in a specific occupation[3]. We focus our analysis on core tasks–tasks which are critical for a specific occupation[4]– within three major occupation groups: Business & Financial Operations Occupations, Computer & Mathematical Occupations and Management Occupations Those encompass a total of 123 different occupations with 2,058 different core tasks, as defined by the 2018 Standard Occupational Classification System (SOC-2018) from the U.S. Bureau of Labor Statistics. We constrain our study to a subset of 149 tasks, where a credible, practical exam could be designed that an LLM could take while interacting only through textual output and without accessing additional tools. In the first part of the results section we report how materials and tools required, remote feasibility and practicality was assessed to systematically identify those tasks.

We test the following 13 different LLMs on our benchmark (the exact API references to the specific model versions are given in parenthesis): GPT-3.5 Turbo (*gpt-3.5-turbo-0125*), GPT-4o (*gpt-4o*), o3 (*o3-2025-04-16*), Gemini 1.5 Flash (*gemini-1.5-flash*), Gemini 2.0 Flash (*gemini-2.0-flash*), Gemini 2.5 Flash (*'gemini-2.5-flash'*), Gemini 2.5 Pro (*gemini-2.5-pro*), Claude 3 Haiku (*claude-3-haiku-20240307*), Claude 3.5 Sonnet (*claude-3-5-sonnet-20240620*), Claude 3.7 Sonnet (*claude-3-7-sonnet-20250219*), Claude Sonnet 4 (*claude-sonnet-4-20250514*), DeepSeek V3 (*openai-api/deepseek/deepseek-chat*) and DeepSeek R1(*deepseek-reasoner*).

For all models except o3, we set the temperature to zero. For o3 we used temperature 1 since, at the date of testing, this was the only setting available for this model. Temperature 0 corresponds to the configuration that is closest to deterministic inference.

4.2 Epoch AI

We obtain publication dates from Epoch AI[5] as well as public releases. For Gemini 2.0 Flash, we recorded a publication date of February 5, 2025, with training compute of 2.43×10^{25} FLOPs [6,7]. For Gemini 2.5 Pro Preview we recorded a publication date of March 25, 2025)[8]

5 Results

We present results in the following order. First we present the results of the LLM's assessment of tools and materials needed to design a meaningful exam for a particular task, the subsequent filtering of tasks and the resulting numbers of successful exams created. Next we report the performance of Claude 3.7 Sonnet, Gemini 2.5 Pro and o3 in creating valid exams. Finally, we report and analyze the performance of 13 different LLMs on these exams.

5.1 Tools, Materials, and Filtering

During the preliminary step of the exam generation pipeline, we requested Claude 3.7 Sonnet to specify a list of tools and materials required to create a meaningful practical, remote exam for a particular task. We then mapped the tools–phrased as tools typically used by humans–to standard tools often integrated with advanced LLMs[9] (see Appendix A.1 for an overview of the mapping). Figure 2 depicts the share of tasks requiring specific combinations of materials or (LLM-)tools for each occupation group. Almost all tasks across all occupation groups require text and data. A large number additionally requires images or virtual labs. When it comes to tools, 225 tasks (i.e. 11%) require no tools, 323 rely on function calling, 409 additionally require computer

[3] `https://www.onetcenter.org/dictionary/20.1/excel/education_training_experience.html`

[4] ONET Center "Task Statements - O*NET 29.3 Data Dictionary at O*NET Resource Center", accessed March 22, 2025, `https://www.onetcenter.org/dictionary/29.2/excel/task_statements.html`

[5] "AI Benchmarking dashboard", accessed May 7, 2025, `https://epoch.ai/data/ai-benchmarking-dashboard`

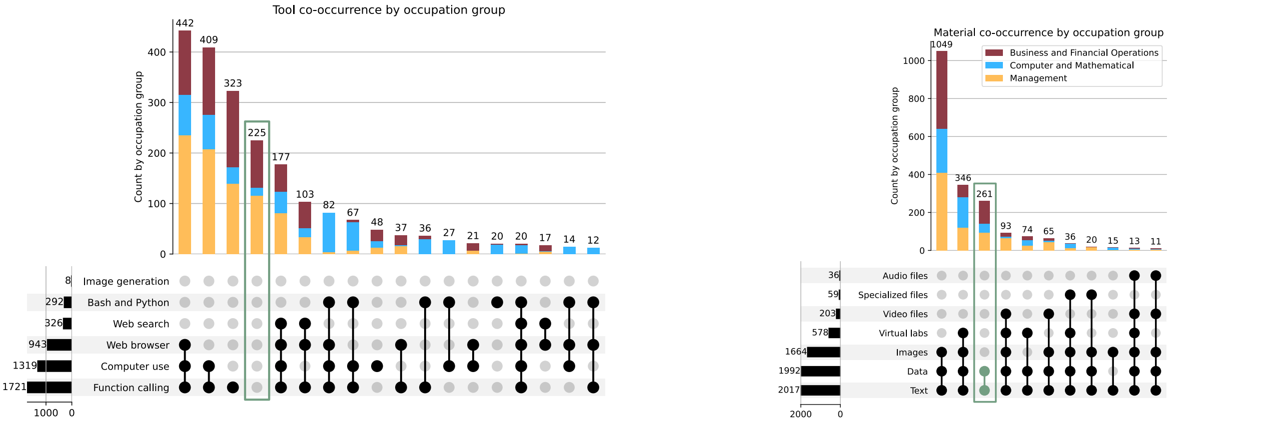
[6] "Gemini 2.0 is now available to everyone", accessed May 7, 2025, `https://blog.google/technology/google-deepmind/gemini-model-updates-february-2025/`

[7] "Gemini 2.5: Google's Revolutionary Leap in AI Architecture, Performance, and Vision", accessed May 7, 2025, `https://ashishchadha11944.medium.com/gemini-2-5-googles-revolutionary-leap-in-ai-architecture-performance-and-vision-c76afc4d6a06`

[8] Gemini API - Release Notes, accessed November 12 2025, `https://ai.google.dev/gemini-api/docs/changelog`

[9] We base our choice and definition of LLM tools on the list of "Standard Tools" provided by the Inspect framework: `https://inspect.aisi.org.uk/tools.html`

use and another 442 are also dependent on access to a web browser. Mostly tasks within Computer and Mathematical Occupations are reliant on Bash and Python.



(a) Tasks requiring specific tools.

(b) Tasks requiring specific materials.

Figure 2: Number of tasks within each occupation group that require generating a specific combination of materials or tools. Only combinations pertaining to at least 10 tasks are depicted. Selected tasks are those highlighted by the green square in the fourth column on panel (a), which require no tool use, and that for materials require no more than data and text in panel (b)

Based on these requirements, we filtered tasks that that could be completed without any additional tools (4th column in Figure 2a) only depended on text and data as materials (marked in light green in Figure 2b). We also excluded tasks which were– according to the LLM's label–not possible to perform remotely or where it wasn't feasible to create a *practical exam*. This resulted in 149 tasks, corresponding to 7.2% of all tasks in the investigated occupation groups. As shown in Table 1, the share of tasks meeting the above defined requirements was largest, within the domain of Business & Financial Operations, with 9.8% and smallest within the domain of Computer & Mathematical Occupations (3.9%).

Appendix A.1 provides a graphical overview of the number of core tasks per occupation as well as the respective share of tasks meeting the selection criteria and the share of valid exams. See Section 3.3 for a description of how a "valid" exam was defined.

	# of tasks	# of examined tasks	% of examined tasks	# of valid exams Claude 3.7 Sonnet	# of valid exams Gemini 2.5 Pro	# of valid exams o3
Business And Financial Operations Occupations	691	68	9.84	61	46	12
Computer And Mathematical Occupations	545	21	3.85	15	18	8
Management Occupations	868	60	6.91	47	52	4

Table 1: Task and Exam Coverage by Occupation Group. This table presents the distribution of tasks and exams across three occupation categories. For each occupation group, we report the total number of core tasks, the subset selected for examination, and the proportion of examined tasks that yielded valid exams by Claude 3.7 Sonnet, Gemini 2.5 Pro and o3.

5.2 Successful exam creation

We attempted to create exams for 149 tasks using three different exam-generator models: Claude 3.7 Sonnet (claude-3-7-sonnet-20250219), Gemini Pro 2.5 (gemini-2.5-pro), and o3 (o3-2025-04-16). Unless otherwise stated, "Claude," "Gemini," and "GPT" refer to these specific model versions. As shown in Table 1, Claude achieved the highest success rate, with 83 % of generated exams passing all validation checks. Our generation pipeline was iteratively developed with Claude, which likely contributed to its strong performance. Gemini passed all checks for 78 % of exams, while GPT achieved only 16 %.

Table 2 summarizes the proportion of exams that failed each validation check. For Claude, the most common failure was the answer key not achieving a perfect score. GPT's primary issue were failures of the self-sense check and for Gemini it was a mixture of both. Manual inspection of several GPT-flagged "senseless" exams confirmed the accuracy of these labels. For instance, one exam tasked the model with creating a test to evaluate whether a *Computer Programmer* could correct errors in a software program by *making appropriate changes and rechecking the program to ensure that the desired results are produced*. However, the corresponding test submission file instructed the exam taker to fill in a "status" for each task and to "set status to 'pass' if ALL its visible tests pass locally, otherwise 'fail'," effectively asking the candidate to self-evaluate rather than actually perform error correction. This inconsistency was detected by our integrated sense check, and the resulting exam was excluded from the valid set. Nevertheless, we cannot rule out the presence of similar errors in other exams marked as valid. Additional illustrative examples of exams flagged as invalid are provided in Appendix A.2.

	Occupation	% of answer keys with score < 99	% of answer keys with score > 100	% of exams relying on fabricated websites	% of exam without separate candidate materials	% of exams not making sense overall	% of exams relying on non-existing image materials	% of exam generations failing during the process
Claude 3.7 Sonnet	Business And Financial Operations Occupations	8.82	0.00	0.00	0.00	1.47	0.00	0.00
	Computer And Mathematical Occupations	28.57	0.00	0.00	4.76	0.00	0.00	0.00
	Management Occupations	16.67	0.00	0.00	3.33	0.00	0.00	0.00
Gemini 2.5 Pro	Business And Financial Operations Occupations	8.00	0.00	6.00	0.00	0.00	0.00	0.00
	Computer And Mathematical Occupations	9.52	0.00	0.00	4.76	9.52	0.00	0.00
	Management Occupations	10.00	0.00	3.33	1.67	10.00	0.00	0.00
o3	Business And Financial Operations Occupations	14.71	0.00	1.47	0.00	69.12	0.00	13.24
	Computer And Mathematical Occupations	27.78	0.00	0.00	0.00	50.00	0.00	28.57
	Management Occupations	18.18	0.00	0.00	0.00	77.27	0.00	1.67

Note: A single exam might fail more than one validation check, thus total sums might exceed numbers reported in Table 1.

Table 2: Validation Failure Analysis by Occupation Group. This table reports the share of attempted exam creations failing due to a specific validation criterion, across occupation categories. Each column represents a distinct validation check.

After investigating the scores of the exam taker models, we further excluded four exams from the subsequent analysis in which the maximum score achieved by any model exceeded 100 or was 0.

5.3 Exam performance

5.3.1 Submission validity

Model	Percentage of Submission Errors	Percentage of Submissions Scoring 0 points	Min. Score	1st Quartile	Median	3rd Quartile	Max. Score
Claude 3 Haiku	36.33	2.34	0.00	0.00	33.00	66.67	100.00
Claude 3.5 Sonnet	43.36	2.73	0.00	0.00	34.50	80.00	100.00
Claude 3.7 Sonnet	6.64	3.12	0.00	52.98	79.41	93.61	100.00
Claude Sonnet 4	1.56	3.12	0.00	60.00	77.56	94.00	100.00
Deep Seek R1	3.91	5.47	0.00	45.09	75.50	92.36	100.00
Deep Seek V3	2.73	5.08	0.00	47.65	71.62	87.62	100.00
GPT 3.5 Turbo	89.45	2.34	0.00	0.00	0.00	0.00	93.00
GPT 4o	48.83	4.30	0.00	0.00	0.00	70.46	100.00
o3	5.47	2.34	0.00	52.30	77.63	91.41	100.00
Gemini 1.5 Flash	9.38	5.47	0.00	31.00	58.65	80.00	100.00
Gemini 2.0 Flash	3.91	5.47	0.00	40.00	68.00	84.78	100.00
Gemini 2.5 Flash	4.69	5.47	0.00	49.80	76.50	93.00	100.00
Gemini 2.5 Pro	12.89	4.69	0.00	41.60	76.00	93.08	100.00

Table 3: Submission Statistics

Only submissions that were valid JSON and adhered to the requested structure were counted as "valid." For most advanced models, only a small fraction of submissions were invalid or scored zero points. As shown in Table 3, across all exams—regardless of the exam generator—GPT-3.5 Turbo exhibited the highest error rate for valid submissions (almost 90%). Manual inspection suggests this was primarily due to the model providing very short responses and not attempting to complete the exam (e.g., "I will now begin working on the Farm and Ranch Financial Management Practical Exam based on the provided data."). Since the erros are mostly driven by the model not completing the task, we drop it from results so that we do not overestimate the improvement in performance.

The second-highest failure rate was observed for GPT-4o. Qualitative review revealed that it frequently produced instructions on how to complete the exam rather than attempting it directly (e.g., "To complete the exam, you need to perform calculations based on the provided data in the 'FarmData.xlsx' file. Here is a step-by-step guide to help you with each task…"). In other cases, it claimed that necessary materials were missing (e.g., "However, since I don't have access to the actual spreadsheet and text file data, I'll guide you through the steps you would take to complete the exam based on the provided instructions."). While we cannot entirely rule out missing materials for some cases, our inspection of sampled examples confirmed that the required files were included in the prompt, indicating that the model's judgment was incorrect. The share of valid submissions that scored zero points was similar across all models and remained below 6 % overall.

5.3.2 Overall model performance

All exams were scored on a range of 0 to 100. As to be expected, considering that most exams were created by Claude 3.7 Sonnet, this model also had the highest median score across all exams (79.41). However, Claude Sonnet 4 (77.56), o3 (77.63), Gemini 2.5 Pro (76.00) and Gemini 2.5 Flash (76.5) performed almost equally well. The top-performing models also achieved similar third quartile scores above 90, demonstrating comparable peak performance capabilities. Claude Sonnet 4 demonstrated the most reliable performance with only 1.56% submission errors and a high first quartile score of 60.00, indicating strong baseline competency across all exam types. Table 3 reports overall performance for each exam taking model, averaged across all occupation groups and exam generator models.

Models show considerable improvement across time. Figure 3 shows the mean performance of different models across the three occupation groups we study. This demonstrates substantial progress in AI capabilities over a relatively short timeframe. Models released in 2024 (Gemini 1.5 Flash, Claude 3 Haiku, GPT-4o, and Claude 3.5 Sonnet) achieved an average performance of 40.5% (SE = 1.16), while models released in 2025 (Gemini 2.0 and 2.5 Flash, Gemini 2.5 Pro, Claude 3.7 Sonnet and Claude Sonnet 4, DeepSeek models, and o3) achieved an average performance of 66.5% (SE = 0.80). This represents a 26 percentage point improvement in just one year, highlighting the rapid pace of advancement in large language model capabilities on workplace tasks.

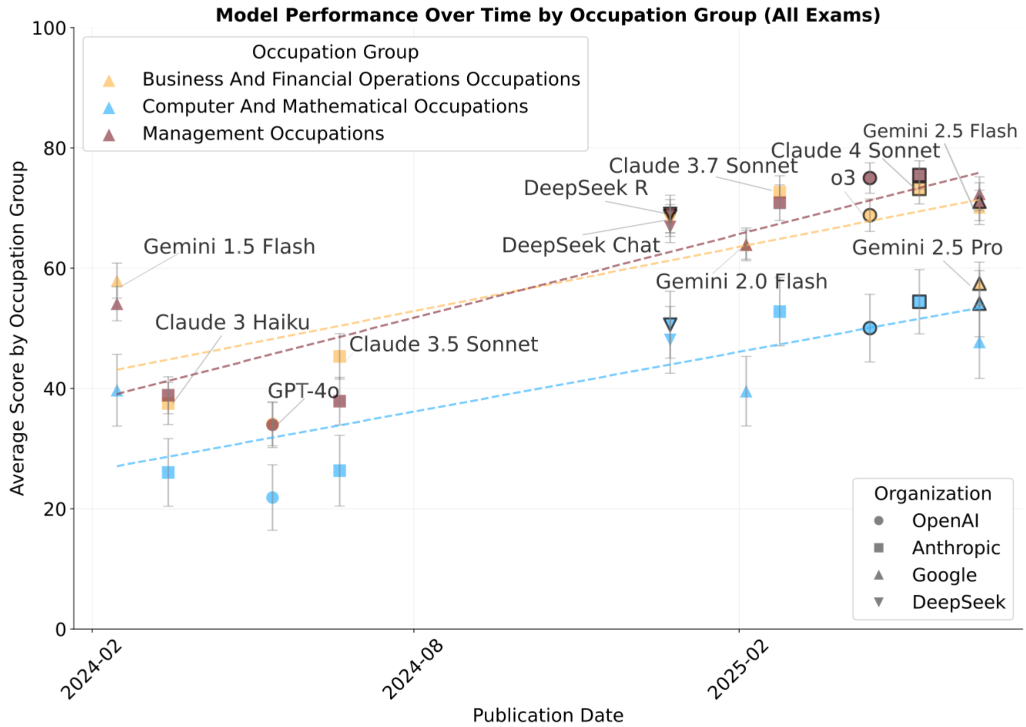


Figure 3: Overall performance of all tested models across the three exam generating models and by occupation groups. The markers with a border indicate a leading model from a particular organization. The error bars report standard errors.

5.3.3 Different exam generator models

Splitting up results according to generating models shows that the patterns observed on the overall level generally also hold for each separately (see Figure 4). We find GPT-3.5 Turbo, GPT 4o and Claude 3 Haiku to be performing worst, mainly due to many submission errors as outlined prior. Notably, the relative performance rankings remain remarkably consistent across all three exam generation models. Claude Sonnet 4 consistently demonstrates above-average performance regardless of which model generated the exams, suggesting robust capabilities that are not dependent on the specific exam format or style. Similarly, o3 shows strong positive deviations across all three generators, particularly excelling on exams created by o3 itself and Claude 3.7 Sonnet. The Gemini model family displays varied performance, with Gemini 2.5 Pro and 2.5 Flash generally performing above average, while earlier versions like Gemini 1.5 Flash show more modest positive deviations. The consistency of these patterns across different exam generators provides evidence that the observed performance differences reflect genuine capability gaps rather than artifacts of specific exam generation approaches. At the level of individual exams, we find some correlations in model performance—that is, the same tasks tend to be easy or difficult for all models (see also Appendix A.4).

Since the exam generator model must "know" the correct answer—having written the exam, grading script, and answer key scoring 100%—one might expect models to display substantial self-preference, performing significantly better on their own exams. We find evidence for this phenomenon, though only to a limited extent. Due to the small number of valid exams generated by o3, we analyze only the 13 tasks where exams were successfully created by all three models, enabling direct comparison across generator-taker combinations.

The results in Table 4 reveal mixed patterns of self-preference. o3 demonstrates the clearest self-advantage, scoring 43.38 on its own exams compared to 26.22 points and 20.84 points by Claude and Gemini, respectively. However, the pattern is less consistent for other models: o3 even performs marginally better on Claude's exams (55.58) than Claude itself (53.77), yet scores highest on Gemini-generated exams (72.46). Similarly, Gemini 2.5 Pro achieves its best performance on Gemini-generated exams (63.23) but does not show a strong self-preference when compared to its performance on Claude-generated exams (43.42). These findings suggest that while self-preference exists, it is not the main driver of performance, and the practical exams are informative of LLMs capabilities.

(see also Appendix A.3).

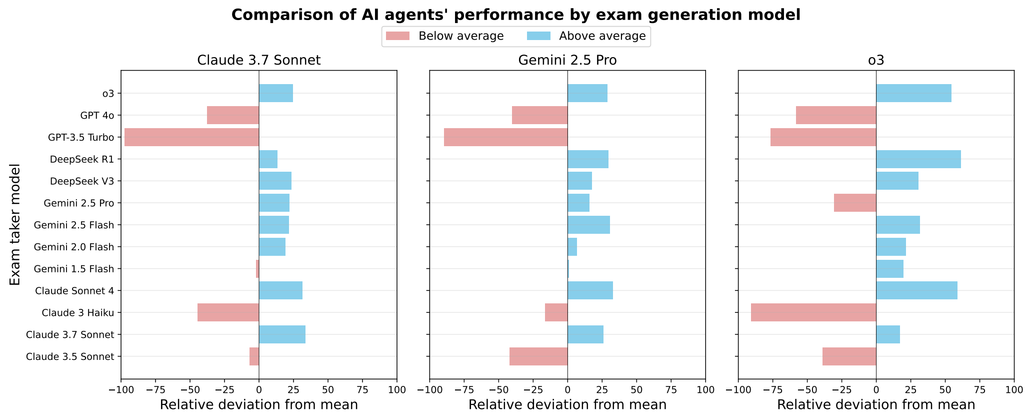


Figure 4: Relative deviation of each exam-taker model's score from the mean score across exams produced by each exam-generator model.

	Exam Taking Model		
	Claude 3.7 Sonnet	GPT o3	Gemini 2.5 Pro
Exam Generating Model			
Claude 3.7 Sonnet	53.77	55.58	43.42
o3	26.22	43.38	20.84
Gemini 2.5 Pro	72.46	67.47	63.23

Table 4: Average performance scores across exam generator-taker model combinations. Values represent mean scores on the subset of 13 tasks where exams were generated by all three models, enabling direct comparison of self-preference effects.

5.3.4 Task difficulty

By studying which tasks LLMs perform better or worse, we can better understand their capabilities. Table 5 shows they struggle most with data analysis and financial calculations, while performing well at document development, classification, and organizational tasks. The hardest tasks center on data manipulation—archival data analysis (6.7% average score), revenue calculations using spreadsheet software (12.9%), and probability table construction (15.9%). In contrast, they achieve high performance on scheduling classes (100%), assisting with budget development (98.1%), and maintaining regulatory knowledge (93.1%).

However, the results are not clearcut between quantitative versus non-quantitative work. Consider two similar tasks: "budget development assistance" (98.1%) versus "calculating revenue and expenses" (12.9%). This comparison shows that LLM performance differs between conceptual understanding and operational execution. Budget development assistance involves strategic thinking, resource allocation discussions, and procedural recommendations, tasks that leverage LLMs' strengths in synthesizing information and providing structured guidance. Revenue calculations, however, require precise numerical operations, potentially complex formulas, that our text-only evaluation may fail.

In other words, our benchmark shows that LLMs can discuss debt liquidation strategies conceptually but struggle to implement the specific calculations those strategies require. To understand the easy and hard task divide we find important words int ask descriptions using TF-IDF and plot them in wordclouds 5. There is an overlap in words between easy and hard tasks. Terms like "budget," "prepare," and "financial" appear in both categories. However, the word cloud shows that "data" appears prominently in difficult tasks, while "development" and "classification" are more prominent in easier ones. This suggest that our benchmark can

Rank	Task Description	Occupation	Broad Category	Avg Score
	Hardest Tasks (Bottom 10%)			
1	Analyze archival data such as birth, death, and disease records.	Biostatisticians	Computer And Mathematical	6.7%
2	Calculate revenue, sales, and expenses, using financial accounting or spreadsheet software.	Online Merchants	Business And Financial Operations	12.9%
3	Construct probability tables for events such as fires, natural disasters, and unemployment, based on analysis of statistical data and other pertinent information.	Actuaries	Computer And Mathematical	15.9%
4	Perform or direct revision, repair, or expansion of existing programs to increase operating efficiency or adapt to new requirements.	Computer Programmers	Computer And Mathematical	17.4%
5	Review employer practices or employee data to ensure compliance with contracts on matters such as wages, hours, or conditions of employment.	Labor Relations Specialists	Business And Financial Operations	18.5%
6	Review clinical protocols to ensure collection of data needed for regulatory submissions.	Regulatory Affairs Specialists	Business And Financial Operations	18.6%
7	Examine budget estimates for completeness, accuracy, and conformance with procedures and regulations.	Budget Analysts	Business And Financial Operations	18.6%
8	Conduct financial modeling for online marketing programs or Web site revenue forecasting.	Search Marketing Strategists	Computer And Mathematical	19.6%
9	Prepare work schedules and station arrangements and keep attendance records.	Gaming Managers	Management	23.7%
10	Prepare data for processing by organizing information, checking for any inaccuracies, and adjusting and weighting the raw data.	Statisticians	Computer And Mathematical	25.0%
11	Correct errors by making appropriate changes and rechecking the program to ensure that the desired results are produced.	Computer Programmers	Computer And Mathematical	26.1%
12	Devise debt liquidation plans that include payoff priorities and timelines.	Personal Financial Advisors	Business And Financial Operations	30.2%
13	Prepare and manage biomass plant budgets.	Biomass Power Plant Managers	Management	30.3%
14	Store, retrieve, and manipulate data for analysis of system capabilities and requirements.	Software Developers, Applications	Computer And Mathematical	30.8%
	Easiest Tasks (Top 10%)			
1	Schedule classes based on availability of classrooms, equipment, or instructors.	Training and Development Specialists	Business And Financial Operations	100.0%
2	Assist with annual budget development.	Advertising and Promotions Managers	Management	98.1%
3	Maintain current knowledge of Equal Employment Opportunity (EEO) and affirmative action guidelines and laws, such as the Americans with Disabilities Act (ADA).	Human Resources Specialists	Business And Financial Operations	93.1%
4	Identify skill development needs for funeral home staff.	Funeral Service Managers	Management	89.9%
5	Recommend changes to company procedures in response to changes in regulations or standards.	Regulatory Affairs Specialists	Business And Financial Operations	89.7%
6	Determine allocations of funds for staff, supplies, materials, and equipment, and authorize purchases.	Education Administrators, Elementary and Secondary School	Management	87.5%
7	Prepare occupational classifications, job descriptions and salary scales.	Compensation, Benefits, and Job Analysis Specialists	Business And Financial Operations	86.6%
8	Prepare and manage distance learning program budgets.	Distance Learning Coordinators	Management	85.6%
9	Analyze new legislation to determine impact on risk exposure.	Risk Management Specialists	Business And Financial Operations	84.6%
10	Review changes to financial, family, or employment situations to determine whether changes to existing debt management plans, spending plans, or budgets are needed.	Credit Counselors	Business And Financial Operations	83.9%
11	Assist in the development of document or content classification taxonomies to facilitate information capture, search, and retrieval.	Document Management Specialists	Computer And Mathematical	82.7%
12	Plan test schedules or strategies in accordance with project scope or delivery dates.	Software Quality Assurance Engineers and Testers	Computer And Mathematical	81.7%
13	Calculate purchase subtotals, taxes, and shipping costs for submission to customers.	Online Merchants	Business And Financial Operations	81.0%
14	Monitor and evaluate the performance of accounting and other financial staff, recommending and implementing personnel actions, such as promotions and dismissals.	Treasurers and Controllers	Management	79.9%

Table 5: Task difficulty based on average scores. At the top the hardest tasks and at the bottom the easiest.

capture the divide between conceptual reasoning about workflows and direct manipulation of quantitative information.

LLMs show substantial improvement in tasks where they currently struggle most. Table 6 shows score increases ranging from 29.58 to 63.33 percentage points between early models (Gemini 1.5 Flash, Claude 3 Haiku, Claude 3.5 Sonnet , GPT-4o) and later models (Claude 3.7 Sonnet, DeepSeek R1, DeepSeek V3, Gemini 2.0 Flash, Gemini 2.5 Flash, Gemini 2.5 Pro, Claude Sonnet 4, o3). The greatest gains occur in tasks requiring both financial understanding and quantitative execution: debt liquidation planning improved by 63.33 percentage points, cost-benefit analysis by 58.55 points, and financial modeling by 47.62 points. Mathematical capabilities are advancing, with biostatistical calculations improving by 46.15 points and tax evaluation by 41.87 points. Improvements extend beyond pure computation to analytical reasoning tasks like logistics data analysis (37.50 points) and campaign performance tracking (33.33 points), indicating LLMs are developing operational skills to execute the strategic insights they can already conceptualize.

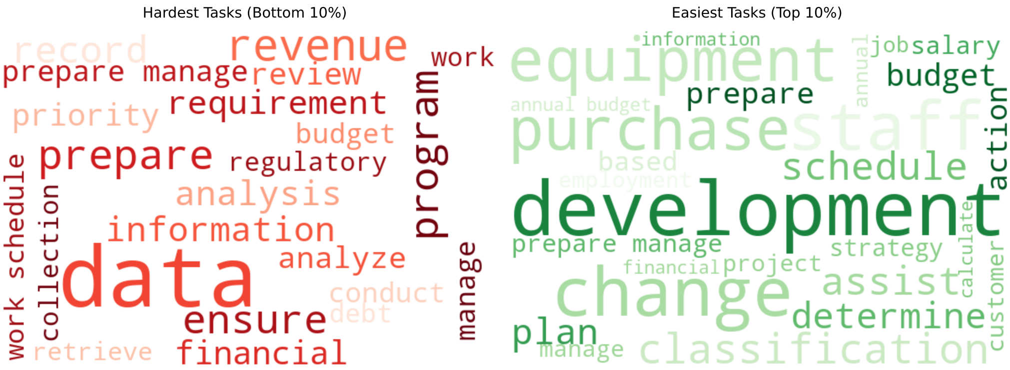


Figure 5: Wordcloud showing most important words (based on TF-IDF) among the hardest and easiest tasks for LLMs.

Rank	Task Description	Occupation	Broad Category	Avg Improvement
1	Devise debt liquidation plans that include payoff priorities and timelines.	Personal Financial Advisors	Business And Financial Operations	63.33
2	Perform cost-benefit analyses to compare operating programs, review financial requests, or explore alternative financing methods.	Budget Analysts	Business And Financial Operations	58.55
3	Conduct financial modeling for online marketing programs or Web site revenue forecasting.	Search Marketing Strategists	Business And Financial Operations	47.62
4	Calculate sample size requirements for clinical studies.	Biostatisticians	Computer And Mathematicals	46.15
5	Evaluate taxpayer finances to determine tax liability, using knowledge of interest and discount rates, annuities, valuation of stocks and bonds, and amortization valuation of depletable assets.	Accountants and Auditors	Business And Financial Operations	41.87
6	Compare liquidity, profitability, and credit histories of establishments being evaluated with those of similar establishments in the same industries and geographic locations.	Credit Analysts	Business And Financial Operations	37.77
7	Analyze or interpret logistics data involving customer service, forecasting, procurement, manufacturing, inventory, transportation, or warehousing.	Logistics Engineers	Business And Financial Operations	37.50
8	Analyze financial data, such as income growth, quality of management, and market share to determine expected profitability of loans.	Credit Analysts	Business And Financial Operations	36.46
9	Track program budgets, expenses, and campaign response rates to evaluate each campaign, based on program objectives and industry norms.	Advertising and Promotions Managers	Management	33.33
10	Prepare staff work schedules and assign specific duties.	General and Operations Managers	Management	29.58

Table 6: Tasks with Greatest Performance Improvement Across Model Generations. This table presents the occupational tasks showing the highest average score improvements between early and latest LLM model versions (Gemini, Claude, and GPT families). For each task, we report the detailed description, associated occupation, broad occupational category, and the percentage point improvement in performance scores.

6 Discussion

This paper addresses the challenge of measuring LLM capabilities in workplace tasks by combining automation exposure measurement from economics and AI benchmarking from computer science. By combining economics' task-based analysis with computer science's benchmarking methods, we develop automated practical exams for O*NET tasks that reveal both current capabilities and improvement trajectories.

Our results indicate that testing full automation via text-only LLMs is currently feasible for only a small subset of tasks—approximately 3-10% across Business & Financial Operations, Management, and Computer & Mathematical occupations. Even within this limited set, the most advanced models achieve median scores between 71-79% on basic exams. Models struggle most with tasks requiring precise numerical computation and data manipulation, while performing better on conceptual reasoning and procedural tasks

We observe substantial improvement over time. Earlier models like GPT-3.5 Turbo struggled to complete the exams in the required format. For the most recent models, these submission failures only account for a minor share of submissions. The task-level analysis reveals that LLMs show the greatest improvement precisely where they currently perform worst. Tasks involving financial calculations, statistical analysis, and quantitative modeling show performance improvements of 30–63 percentage points across model generations.

Methodologically, this study brings us closer to understanding about real-world LLM capabilities in a systematic way. Our approach suggests it may be possible for LLM systems to create benchmarks that say something meaningful about LLMs own capabilities. The alignment between observed performance patterns and expected capabilities, combined with sensible improvement trajectories, suggests that synthetic benchmark generation may provide a scalable approach to capability assessment.

Our work has several limitations. Since we use LLMs to evaluate other LLMs, there is a risk of creating a self-referential evaluation system disconnected from real-world performance. While we implemented sanity checks and observe consistent patterns across different exam generators, validation against human performance is missing to have a reliable benchmark. Our implementation focuses only on tasks that can be evaluated through text-only interaction without external tools or multimedia inputs, which excludes a significant portion of workplace tasks. Moreover, as documented in Appendix A.2 the LLM generated exams are not perfect, adding noise to the capabilities measure.

Several promising directions could extend this research. Manual examination of a random sample of exams by human experts could validate that the evaluations are meaningful and free from hallucinations. More comprehensively, comparing LLM performance against human workers (e.g., by recruiting freelancers to complete the same exams) would establish a baseline for automation potential. Additionally, one could consider using public data or exam data from universities (see e.g. Fan et al., 2025) to verify some of the tasks.

Extending beyond text-only models to incorporate multimodal capabilities (e.g., processing images and audio) and tool use (e.g., executing code, manipulating spreadsheets) would increase task coverage, potentially including many of the 90–95% of tasks currently excluded. We are currently working on integrating tool use into the framework. On the evaluation side, we could further extend to tasks requiring non-numeric evaluation by adopting a "LLM-as-a-judge" methodology (e.g Gu et al. (2025)). Moreover, exploring the relationship between LLM automation potential and task desirability could reveal whether AI progress disproportionately affects tasks typically viewed as unpleasant or tedious by humans or tasks they find engaging–an important consideration for labor market policies and workforce planning.

A key advantage of having a systematic benchmark that can be easily updated is its potential for technological forecasting and model backtesting. Drawing on the technological forecasting literature (François Lafond 2025), approaches like Wright's Law and Moore's Law have demonstrated predictive accuracy in capturing how technologies improve over time (Wright 1936; Farmer and Francois Lafond 2016; François Lafond et al. 2018). For LLMs specifically, scaling laws have established predictable relationships between model parameters, training data, and performance metrics (Kaplan et al. 2020; Hoffmann et al. 2022). Therefore, this benchmark could feed into statistical models that inform future automation potential with quantifiable uncertainty bounds.

The ability to systematically measure AI capabilities across workplace tasks creates opportunities for adaptive complementarity between human workers and AI systems. By tracking which tasks AI masters and at what rate, this measurement framework can enables both educational institutions and AI developers to steer toward complementarity. Education systems can shift focus toward tasks where human expertise retains value, while AI developers can prioritize automating tasks workers find tedious rather than those that provide meaning or better career prospects Shao et al. 2025; Petrova et al. 2025. As Autor and Thompson 2025 show, automation's wage effects depend on whether it eliminates expert or inexpert tasks from occupations. Tracking AI capabilities allows stakeholders to anticipate and potentially avoid wage stagnation by ensuring AI development creates new domains for human expertise rather than simply eliminating existing ones. This dynamic measurement system updates as both AI models and job requirements evolve, transforming technological forecasting from speculation to empirical tracking that informs education policy and development strategy.

7 Acknowledgements

We acknowledge support from The Brookings Institution. We are also grateful to Simon Bunel, Simone Daniotti, Harald Fadinger, Iyngkarran Kumar, Francois Lafond, Sam Manning, Frank Neffke, and Gregor Schubert for their feedback and discussions on this work. We also acknowledge Open AI token credit funding.

References

Aghion, Philippe et al. (2025). "How different uses of AI shape labor demand: evidence from France". In: *AEA Papers and Proceedings*. Vol. 115. American Economic Association 2014 Broadway, Suite 305, Nashville, TN 37203, pp. 62–67.

Autor, David and Neil Thompson (2025). "Expertise". In: *Journal of the European Economic Association*, jvaf023.

Chang, Yupeng et al. (2023). *A Survey on Evaluation of Large Language Models*. arXiv: 2307.03109 [cs.CL]. URL: https://arxiv.org/abs/2307.03109.

Department for Education (DfE) London, UK (2024). *Generative AI in education: Educator and expert views*. https://www.gov.uk/government/publications/generative-ai-in-education-educator-and-expert-views.

Eloundou, Tyna et al. (June 2024). "GPTs are GPTs: Labor market impact potential of LLMs". In: *Science* 384.6702, pp. 1306–1308. DOI: 10.1126/science.adj0998.

Fan, Yu et al. (2025). *LEXam: Benchmarking Legal Reasoning on 340 Law Exams*. arXiv: 2505.12864 [cs.CL]. URL: https://arxiv.org/abs/2505.12864.

Farchi, Eitan et al. (2024). "Automatic Generation of Benchmarks and Reliable LLM Judgment for Code Tasks". In: *arXiv preprint arXiv:2410.21071*.

Farmer, J Doyne and Francois Lafond (2016). "How predictable is technological progress?" In: *Research Policy* 45.3, pp. 647–665.

Felten, Edward, Manav Raj, and Robert Seamans (2021). "Occupational, industry, and geographic exposure to artificial intelligence: A novel dataset and its potential uses". In: *Strategic Management Journal* 42.12, pp. 2195–2217. DOI: `https://doi.org/10.1002/smj.3286`. eprint: `https://sms.onlinelibrary.wiley.com/doi/pdf/10.1002/smj.3286`. URL: `https://sms.onlinelibrary.wiley.com/doi/abs/10.1002/smj.3286`.

— (2023). *How will Language Modelers like ChatGPT Affect Occupations and Industries?* arXiv: `2303.01157 [econ.GN]`. URL: `https://arxiv.org/abs/2303.01157`.

Frank, Morgan R, Yong-Yeol Ahn, and Esteban Moro (2025). "AI exposure predicts unemployment risk: A new approach to technology-driven job loss". In: *PNAS nexus* 4.4, pgaf107.

Frey, Carl Benedikt and Michael A Osborne (2017). "The future of employment: How susceptible are jobs to computerisation?" In: *Technological forecasting and social change* 114, pp. 254–280.

Golchin, Shahriar et al. (2024). *Large Language Models As MOOCs Graders*. arXiv: `2402.03776 [cs.CL]`. URL: `https://arxiv.org/abs/2402.03776`.

Gu, Jiawei et al. (2025). *A Survey on LLM-as-a-Judge*. arXiv: `2411.15594 [cs.CL]`. URL: `https://arxiv.org/abs/2411.15594`.

Guha, Neel et al. (2023). "LegalBench: A collaboratively built benchmark for measuring legal reasoning in large language models". In: *Advances in Neural Information Processing Systems* 36, pp. 44123–44279.

Hauser, Jakob et al. (2024). "Large Language Models' Expert-level Global History Knowledge Benchmark (HiST-LLM)". In: *Advances in Neural Information Processing Systems* 37, pp. 32336–32369.

Hendrycks, Dan et al. (2021). "Measuring mathematical problem solving with the MATH dataset". In: *arXiv preprint arXiv:2103.03874*.

Hoffmann, Jordan et al. (2022). "Training compute-optimal large language models". In: *arXiv preprint arXiv:2203.15556*.

Kaplan, Jared et al. (2020). "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361*.

Kim, Seungone et al. (2023). "Prometheus: Inducing fine-grained evaluation capability in language models". In: *The Twelfth International Conference on Learning Representations*.

Lafond, François (2025). *Forecasting technological progress*. Tech. rep. Institute for New Economic Thinking at the Oxford Martin School, University of Oxford.

Lafond, François et al. (2018). "How well do experience curves predict technological progress? A method for making distributional forecasts". In: *Technological Forecasting and Social Change* 128, pp. 104–117.

Lichtinger, Guy and Seyed Mahdi Hosseini Maasoum (2025). "Generative AI as Seniority-Biased Technological Change: Evidence from US Résumé and Job Posting Data". In: *Available at SSRN*.

Liu, Mingxin et al. (2024). "Performance of ChatGPT across different versions in medical licensing examinations worldwide: systematic review and meta-analysis". In: *Journal of Medical Internet Research* 26, e60807.

Miserendino, Samuel et al. (2025). *SWE-Lancer: Can Frontier LLMs Earn* 1$Million from Real-World Freelance Software Er$ arXiv: `2502.12115 [cs.LG]`. URL: `https://arxiv.org/abs/2502.12115`.

Neffke, Frank et al. (2024). *Economic Complexity Analysis*. Tech. rep. Utrecht University, Department of Human Geography and Spatial Planning ...

Panickssery, Arjun, Samuel Bowman, and Shi Feng (2024). "LLM evaluators recognize and favor their own generations". In: *Advances in Neural Information Processing Systems* 37, pp. 68772–68802.

Petrov, Ivo et al. (2025). *Proof or Bluff? Evaluating LLMs on 2025 USA Math Olympiad*. arXiv: `2503.21934 [cs.CL]`. URL: `https://arxiv.org/abs/2503.21934`.

Petrova, Maria et al. (2025). "Career Values for Labor Markets: Evidence from Robot Adoption". In.

Rio-Chanona, R Maria del et al. (2025). "AI and jobs. A review of theory, estimates, and evidence". In: *arXiv preprint arXiv:2509.15265*.

Saxon, Michael et al. (2024). *Benchmarks as Microscopes: A Call for Model Metrology*. arXiv: `2407.16711 [cs.SE]`. URL: `https://arxiv.org/abs/2407.16711`.

Schoenegger, Philipp et al. (2024). "Wisdom of the silicon crowd: LLM ensemble prediction capabilities rival human crowd accuracy". In: *Science Advances* 10.45, eadp1528.

Shao, Yijia et al. (2025). "Future of Work with AI Agents: Auditing Automation and Augmentation Potential across the US Workforce". In: *arXiv preprint arXiv:2506.06576*.

Styles, Olly et al. (2024). "WorkBench: a benchmark dataset for agents in a realistic workplace setting". In: *arXiv preprint arXiv:2405.00823*.

Teutloff, Ole et al. (2025). "Winners and losers of generative AI: Early Evidence of Shifts in Freelancer Demand". In: *Journal of Economic Behavior & Organization*, p. 106845.

Wang, Alex, Yada Pruksachatkun, et al. (2019). "SuperGLUE: A stickier benchmark for general-purpose language understanding systems". In: *Advances in neural information processing systems* 32.

Wang, Alex, Amanpreet Singh, et al. (2018). "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461*.

Webb, Michael (2019). "The impact of artificial intelligence on the labor market". In: *Available at SSRN 3482150*. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3482150.

Wright, Theodore P (1936). "Factors affecting the cost of airplanes". In: *Journal of the aeronautical sciences* 3.4, pp. 122–128.

Xu, Frank F et al. (2024). "The Agent Company: benchmarking LLM agents on consequential real world tasks". In: *arXiv preprint arXiv:2412.14161*.

Zheng, Lianmin et al. (2023). *Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena*. arXiv: 2306.05685 [cs.CL]. URL: https://arxiv.org/abs/2306.05685.

Appendix A Additional results

A.1 Material and tools

In this section we expand on our results on materials and tools required for each task. Table 7 we show the correspondence table we designed to map human tools into LLM tools using the Inspect AI framework. In Figure6 we show the results of the labeling with Claude 3.7 Sonnet of the required materials and tools for all core tasks in the selected occupations groups. After using material and tools to filter tasks, only a minority where kept for evaluation, Figure 10 shows the number of core tasks per occupation in green and in red the ones selected for evaluation.

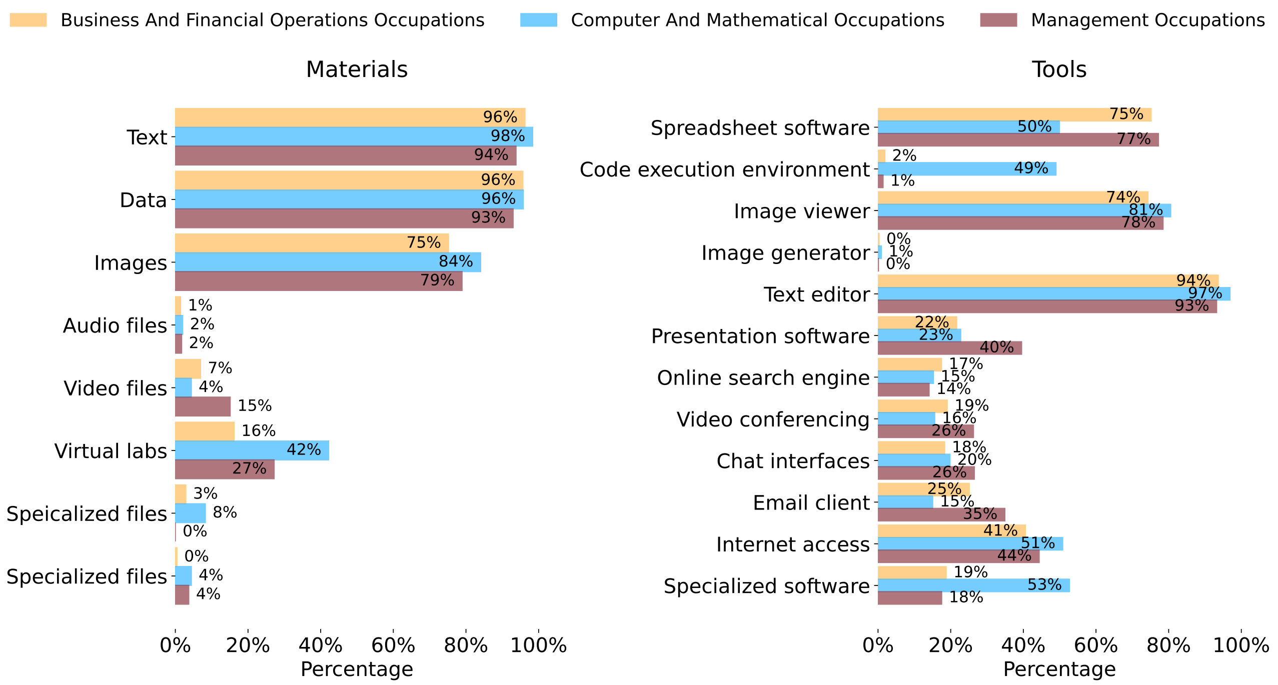


Figure 6: Percentage of tasks within each occupation group that require generating a specific material or tool for the practical exam as labeled by Claude 3.7 Sonnet.

Table 7: Mapping of human tools to standard tools commonly built into advanced LLMs (based on the Basic tools available in the Inspect AI framework: `https://inspect.aisi.org.uk/tools.html`)

Human tools	**Corresponding LLM tool**
Spreadsheet software	No tool usage
Code execution environment	Bash and Python
Image viewer	Multimodal
Image generator	No tool but image generation model
Text editor	No tool usage
Presentation software	Computer use
Online search engine	Web search
Video conferencing	Currently out of scope
Chat interfaces	Currently out of scope
Email client	Web browser
Specialized software	Computer use

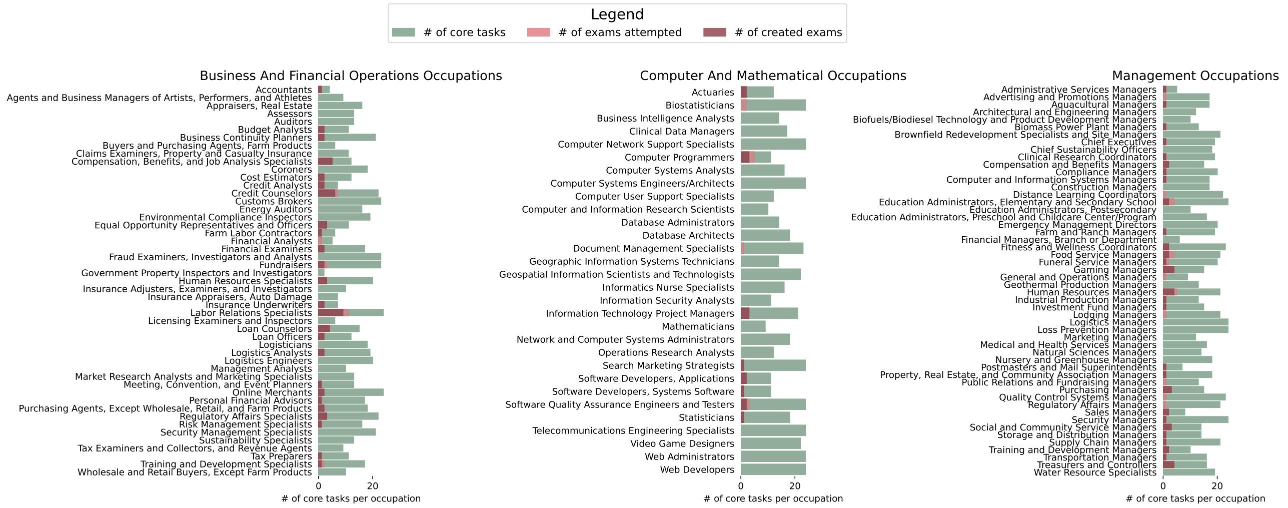


Figure 7: Number of core tasks per occupation group, share of tasks matching our filter criteria and share of valid exams produced by Claude 3.7 Sonnet

A.2 Sense check failures o3

For the exam generating model o3 the most relevant reason for failing the validation checks was exams being marked as making no sense. We manually examined a few examples of those instances, which confirmed that the model was correct in labeling those as invalid as well as pointing out the correct reasons for doing so. Some examples of the reasoning provided by the model:

- Overall the exam is well-designed and practical: it mirrors an LR specialist's 90-minute task, uses realistic 2025 data, and gives clear JSON instructions. The grading script largely matches the rubric and key. HOWEVER a critical flaw exists: the narrative summary is a required deliverable, yet the scoring logic awards only 4 points for it and the quick-pass rule ignores it completely. A candidate can submit a totally empty "narrative_summary" (earning 0/4 on that sub-item), still score 80, and therefore pass while missing a critical part of the assignment. This violates check 4 ("no scenario where a candidate can pass overall despite failing a critical part"). Suggestion: make the narrative worth enough points that omitting it prevents reaching the 80-point pass line, or add a hard fail if the field is blank. Minor ambiguities: (a) risk_scenarios list size—schema says "up to two", but sample shows two objects; script tolerates one, so wording could confuse candidates. (b) Key order is said to be mandatory for candidates, yet the script only checks presence, not order. Otherwise the exam content, materials, and answer key align and the key would score 100 % on the script.
- The exam is practical and the tasks and materials are realistic for a 2025 Labor-Relations Specialist (checks 1 & 2 pass). Instructions and schema are mostly unambiguous, though baseline-year for cost calculations could be clearer (check 3 mostly passes). However the grading script auto-scores the core 'proposed_text' quality only by textual similarity (0.6 Jaccard/Sequence = 10 pts) with no substantive or legal validation. A candidate could change the bereavement clause to two days (illegal) yet still share >90 % token overlap and receive full points, potentially passing with a non-compliant result. Because a critical failure (legal non-compliance) can still earn a passing score, check 4 fails. The key does earn 100 %, but the rubric-vs-script gap is a structural weakness that invalidates the reliability of the exam.
- The core task is practical and realistic for a 2025 fundraising analyst, and the answer key and grader are internally coherent; however, the candidate-facing instructions are contradictory. The first JSON schema presented to the candidate (early in the brief) uses keys such as `"chosen_tactic_code"`, whereas the later, more detailed *Answer & File-format Requirements* section and the grading script demand keys prefixed with `"Q1_..."`, `"Q2_..."`, and so on. A candidate following the earlier schema would submit valid work yet be auto-failed by the grader for missing keys. This ambiguity violates check #3. Fix by showing only one canonical schema (the Q-prefixed one) throughout, or by modifying the grading script to accept both.

A.3 Scores by occupation and exam generator model

In the main text we presented the evolution of LLM capabilities across time excluding GPT-3.5 Turbo since for most tasks it was not able to provide valid output. For completeness in Figure 8 we include GPT-3.5 Turbo results. However, we note that in previous iterations of our work GPT-3.5 Turbo had achieved roughly 30%

accuracy. In Figure 9 we show model performance of all models ordered by family and in Figure 10 correlation in performance between models.

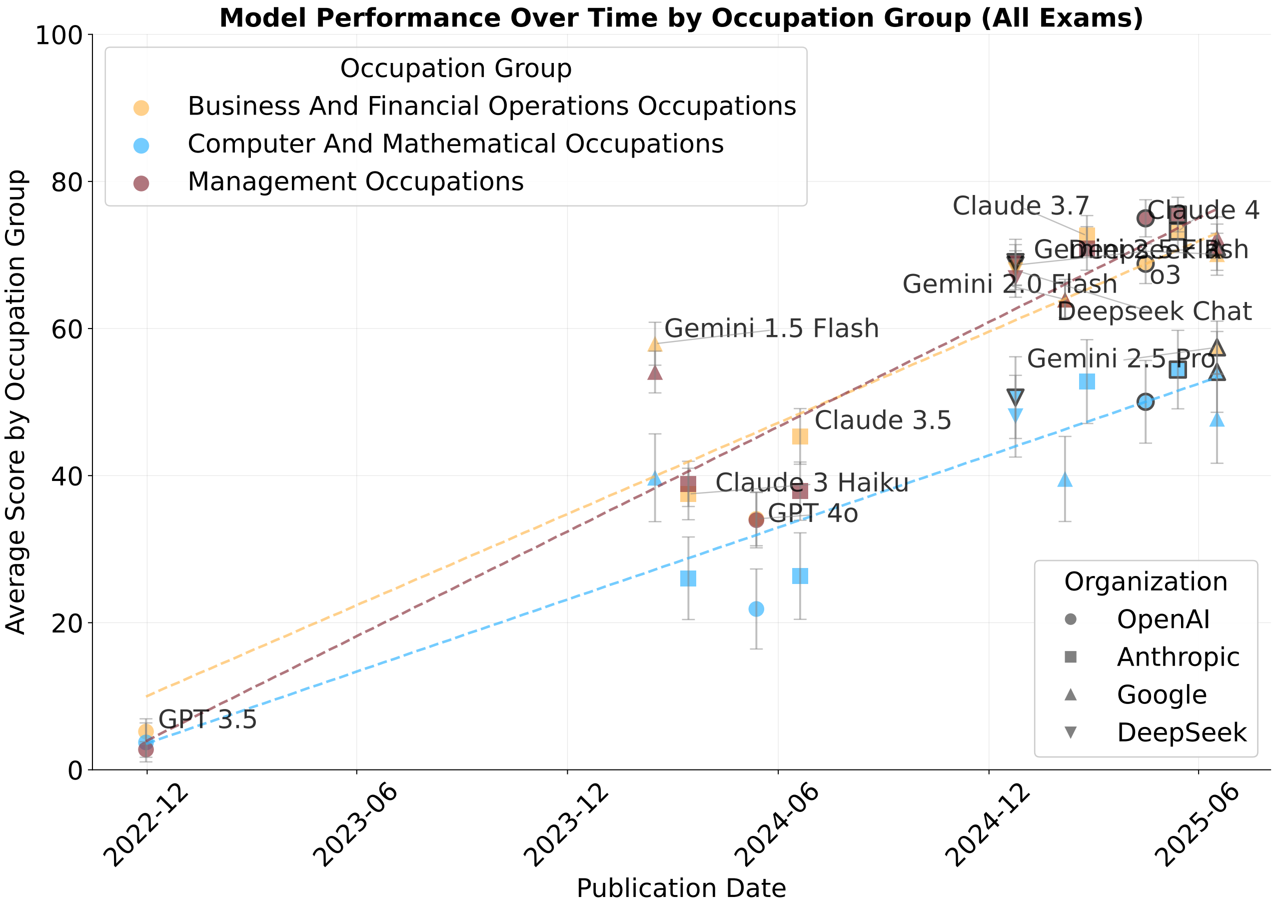
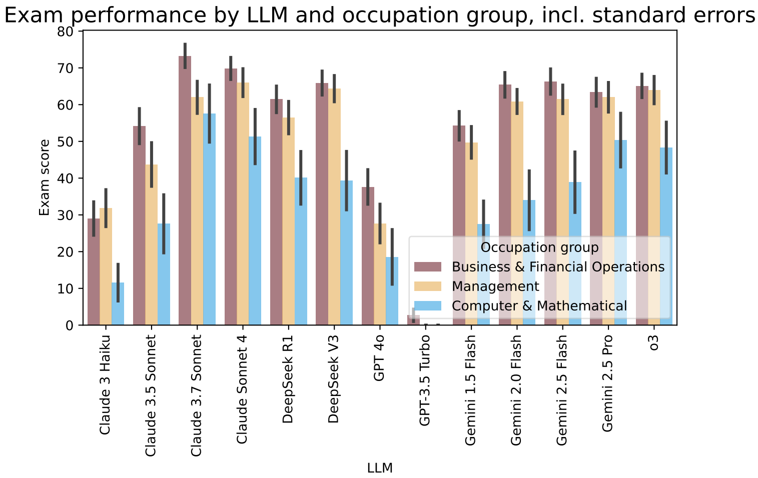
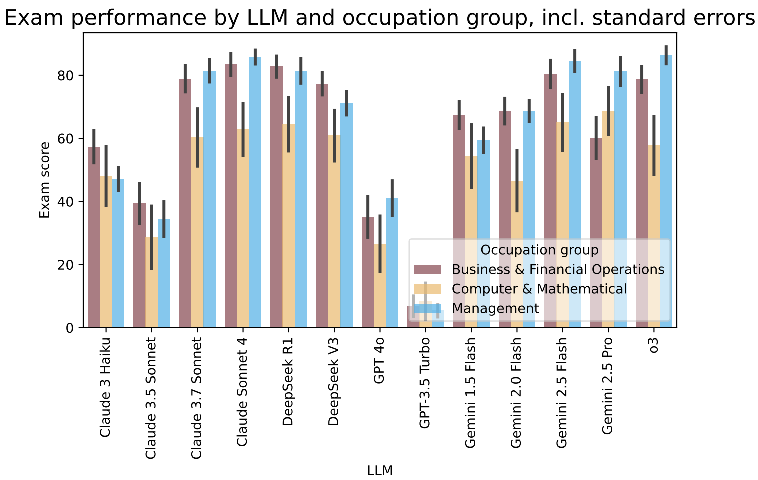


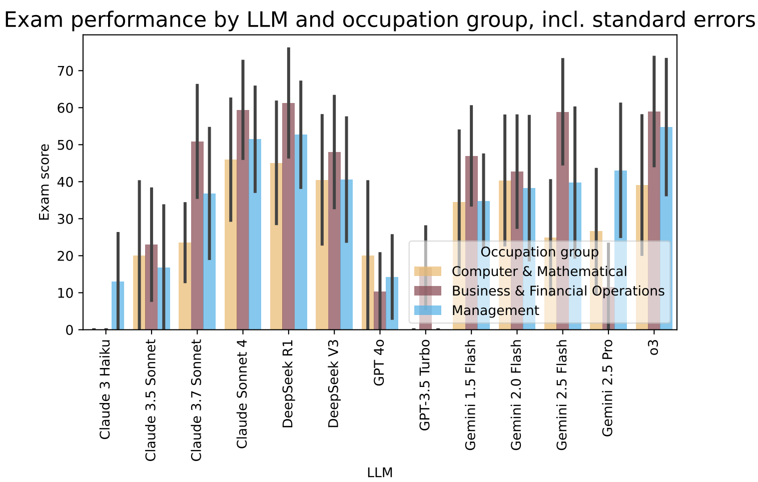
Figure 8: Mean performance of models across time for the three occupation groups. We include GPT-3.5 Turbo here too.



(a) Exam generator Claude 3.7 Sonnet



(b) Exam generator Gemini 2.5 Pro



(c) Exam generator o3

Figure 9: Mean performance of all tested models across occupation groups. Error bars denote standard error across a major occupation group

A.4 Correlations of model performance

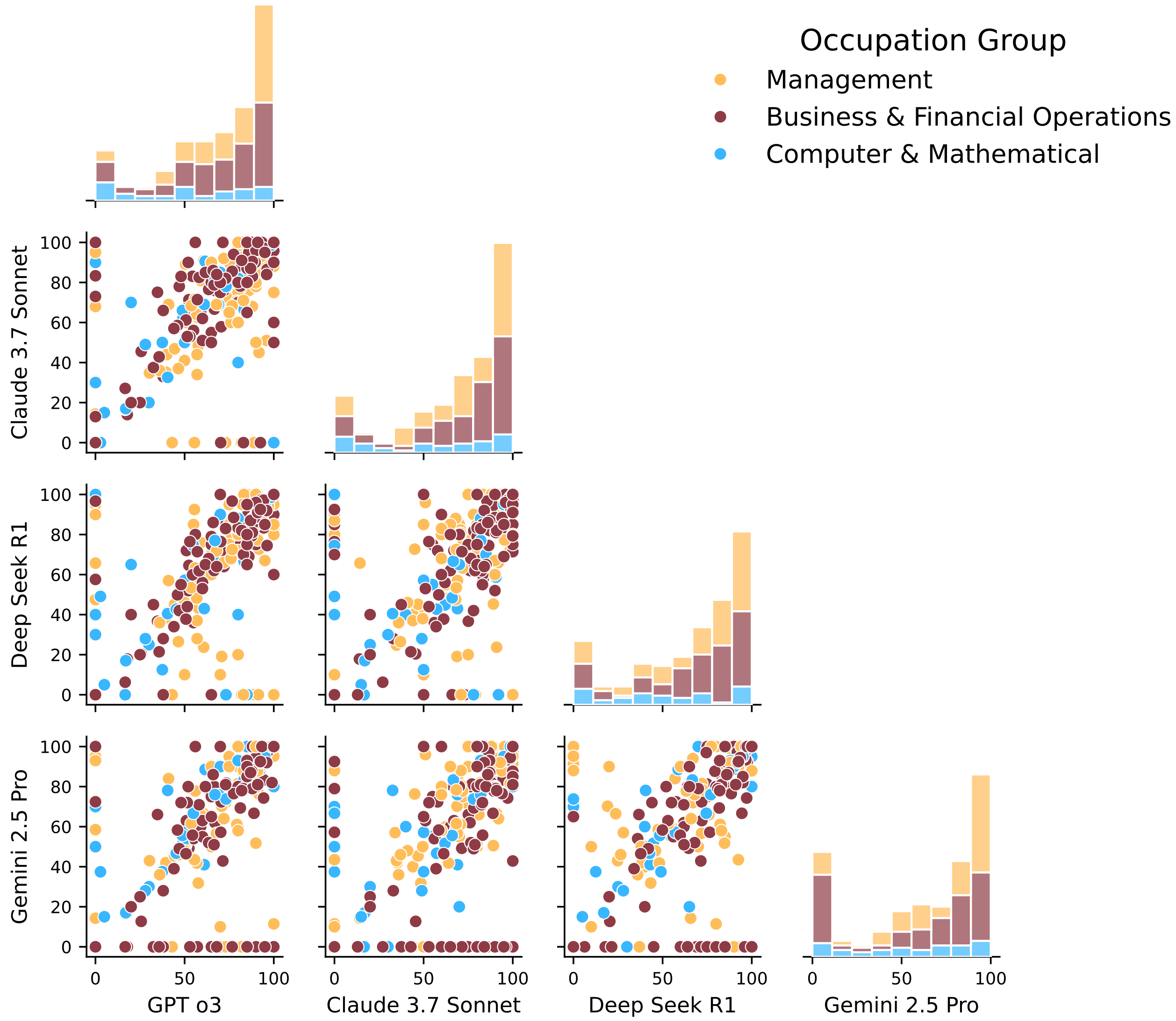


Figure 10: Correlation of exam scores of various state of the art models.

Appendix B Prompts

B.1 Required materials and tools

The lists of required materials and tools were generated for each task individually using prompts 1 and 2, with the following fill ins:

- *occupation*: Occupation the task belongs to according to O*NET definition.
- *task_description*: One sentence description of the task from the O*NET database.
- *task_id*: Unique identifier for each task.

Prompt 1: System prompt for generating list of required tools and materials.

```
You are an excellent examiner of {occupation}'s capabilities.
The overall objective is to evaluate, via a **practical** online exam with a total
    duration of 90min, whether {occupation} can {task_description}.
```

Prompt 2: User prompt for generating list of required tools and materials.

```
Task ID: {task_id}

Your assignment is to determine if (1) the task can be performed remotely, (2) it is
    possible to design a meaningful, **practical** exam for this task and (3) whether
    such an exam can be evaluated using fact-based scoring. Further, you should identify
    the required tools and materials that exam takers need to use.

**Definitions:**
- **Remote feasibility**: A task can be performed remotely if it doesn't require a person
     to be at a specific place or interact with humans face-to-face.

- **Tools:** Software or applications (e.g. coding & development tools, text editor,
    presentation software, image generator), that are **absolutely** critical for an exam
     taker to complete any practically relevant and meaningful exam for this task.

- **Materials:** Digital content (e.g., data files, text, images, audio files) that form
    part of the test content.

- **Fact based scoring:** Whether the exam can be evaluated using a predefined answer key
     (objective scoring) or requires subjective assessment, such as evaluating the
    quality of written responses or creative work.

- **Practical exam:** A practical exam is a an exam actually testing whether the
    described task can be performed successfully. An exam testing the knowledge about the
     task is NOT a practical exam.

- **Virtual lab:** A virtual lab is an interactive digital environment that simulates
    real-world settings, allowing users to perform tasks, collaborate with others, and
    test scenarios remotely.

- **Images:** Such as product images, photographs, maps, graphs and plots, etc.

- **Specialized software:** Specialized software refers to computer programs designed for
     specific industry tasks or technical functions, such as Simulink for system modeling
    , CAD for 3D design or SAP for enterprise resource planning. This category should be
    used if there is no way to solve the exam using combinations of the other tools and
    software listed under tools, but some more specialized software is required.

- **Specialized files:** Specialized files refers to files that cannot be parsed into
    text, data, images, audio or video but are also not part of a virtual lab. This
    category should be used if there is no way to provide the necessary exam materials
    using a combination of the other materials listed.

**Instructions:**
1. **Remote Feasibility:**
   Evaluate whether {task_description} can be performed online/remotely or if it requires
        in-person presence.
   - **If in-person required:** Output `"can_be_performed_remotely": false` and set all
       other fields (tools and materials) to `"NA"`.
   - **If remote:** Output `"can_be_performed_remotely": true` and continue with the
       evaluation.

2. **Materials Required:**
   For each material type, determine if it is necessary to provide materials of this type
        as part of an exam to evaluate {occupation}'s ability to perform the task ({
       task_description}).
   Options are: "Required"or "Not required"
   The materials include:
   - "Text"
   - "Data"
   - "Images"
   - "Audio files"
   - "Video files"
   - "Virtual labs"
   - "Specialized files"
```

```
3. **Tools Required:**
   For each tool, assess whether it is needed to complete an exam that evaluates whether
       an individual can do {task_description}.
  Options are: "Required" or "Not Required".
  The tools include:
   - "Spreadsheet software"
   - "Code execution environment"
   - "Text editor"
   - "Image viewer"
   - "Image generator"
   - "Presentation software"
   - "Online search engine"
   - "Video conferencing"
   - "Chat interfaces"
   - "Email client"
   - "Internet access"
   - "Specialized software"


4. **Feasibility of a practical exam:**
    Evaluate whether the task can meaningfully be tested in a practical, remote exam.
    - If you think this is possible, answer True,
    - Otherwise answer False

5. **Chain-of-Thought Reasoning:**
   Optionally, include a brief chain-of-thought explanation (no more than 150 words) for
       your evaluations in a field called `"chain_of_thought"`.

**Output Requirement:**
Return a JSON object strictly adhering to the provided structure, without any extra
    commentary outside of the JSON fields.

**Expected JSON Structure:**
{{
  "task_id": "{task_id}",
  "occupation": "{occupation}",
  "task_description": "{task_description}",
  "can_be_performed_remotely": true/false,
   "feasibility_practical": true/false,




  "materials": {{
    "Text": "Not Required/Required/NA",
    "Data": "Not Required/Required/NA",
    "Images": "Not Required/Required/NA",
    "Audio files": "Not Required/Required/NA",
    "Video files": "Not Required/Required/NA",
    "Virtual labs": "Not Required/Required/NA",
    "Specialized files":"Not Required/Required/NA",
  }},

   "tools": {{
   "Spreadsheet software": "Not Required/Required/NA",
   "Code execution environment": "Not Required/Required/NA",
   "Image viewer": "Not Required/Required/NA",
   "Image generator": "Not Required/Required/NA",
   "Text editor": "Not Required/Required/NA",
   "Presentation software": "Not Required/Required/NA",
   "Online search engine": "Not Required/Required/NA",
   "Video conferencing": "Not Required/Required/NA",
   "Chat interfaces": "Not Required/Required/NA",
   "Email client": "Not Required/Required/NA",
```

```
    "Internet access": "Not Required/Required/NA",
    "Specialized software:""Not Required/Required/NA",
  }},
  "chain_of_thought": "Brief explanation (no more than 150 words)."
}}
```

B.2 Exam creation pipeline

B.2.1 System prompt

For each task a series of iterative prompts was used to create the exam.

- *occupation*: Occupation the task belongs to according to O*NET definition.
- *task_description*: One sentence description of the task from the O*NET database
- *tools_list*: List of tools required for the exam.
- *materials_list*: List of types of materials that can be given to the candidate.

Prompt 3: System prompt for exam generation

```
You are an excellent examiner of {occupation} capabilities. Design a remote, **practical
    ** exam to verify whether a {occupation} can {task_description}.
This exam will have two parts (basic and advanced).

- Basic: A basic practical exam should assess mostly whether an individual has the
    minimum required knowledge and skill to perform the task to a satisfactory standard
    in a junior or entry-level role. It should be complex enough to resemble a real world
     task, but simple enough that a qualified {occupation} at entry level should score
    more than 80%.
- Advanced: An advanced practical exam should asses whether an individual demonstrates a
    high degree of proficiency in performing the task independently and effectively in a
    senior or expert role. An established person in the profession should be able to
    score 80% or more, but an entry level {occupation} may struggle to get more than 50%
    correct.

Your current task is **only** to design the {level} exam.

Take into account that typically more than half of positions on this occupation require
    at least a {education}.

### Context
{tools_instructions}
{materials_instructions}
- Design a **practical** exam that can be completed remotely using only these tools. A
    practical exam is an exam actually testing whether the described task can be
    performed successfully. An exam testing knowledge about the task is NOT a practical
    exam.
- To simplify evaluation, the candidate should submit answers in a structured JSON format
    . Name the file "test_submission.json".
- The candidate should be able to complete the exam in maximum 90 minutes.
```

B.2.2 Exam generation

The exam generation happened in six steps, each consecutive prompt including the agent's response to all previous prompts.

Prompt 4: First prompt in the exam generation pipeline: Exam overview

```

### Your assignment:
Provide an overview of the exam's purpose and optimal structure for the exam creator and
    exam evaluator, emphasizing how to make this a practical rather than theoretical
    assessment.  Highlight any strategic elements in the exam -designsuch as multi-step
    processes, data relationships, or decision -pointsthat will differentiate candidates
    who can actually perform the task from those who only understand it conceptually.
    Outline any trick questions or structures in the materials that will be useful to
    know for the exam generator and evaluator.
```

Prompt 5: Second prompt in the exam generation pipeline: Instructions

```
Here is brief explanation of the exam's purpose and structure intended for the evaluator:
     <examoverview> {answer_overview} </examoverview>

    ### Your assignment:

    Based on the explanation write clear, concise instructions for the candidate
        including:
    - What they need to accomplish (without prescribing specific methods)
    - Brief description of any materials that will be provided
    - Expected format for answer submission
    - The actual test they need perform, i.e. the tasks that need to be done or questions
         that need to be answered.

    IMPORTANT: When designing the test, eliminate any opportunities for candidates to
        make arbitrary choices (like custom account codes, naming conventions, or
        classification systems) that would complicate evaluation. Either:
    - Provide pre-defined structures/codes that must be used, or
    - Design questions with objectively verifiable numerical/text answers that don't
        depend on the candidate's approach.
    - You can ask for text answers that can be compared to an exact match, but avoid
        asking for text answers such as justification that require interpretation and/or
        with many possible correct answers.
```

Prompt 6: Third prompt in the exam generation pipeline: Material generation

```
Here is brief explanation of the exam's purpose and structure intended for the evaluator:
     <examoverview> {answer_overview}</examoverview>
    Here are the instructions for the candidate: <instructions> {answer_instructions} </
        instructions>

    ## Your assignment:
    - If the exam doesn't require any additional material, just respond with "No material
         required".
    - Otherwise, create two parts:
    1. Synthetic test materials (CSV contents, datasets, etc.) that have predictable
        outcomes. Include the actual content to be provided to candidates and ensure all
        materials have clear identifiers, labels, or pre-defined categories that prevent
        ambiguity.
    2. An explanation for the evaluator on how these materials were created and any
        knowledge helpful for knowing the correct answers

    Format your response with these specific XML tags:
    <MATERIALS_FOR_CANDIDATE>
    [Include here the actual content to be provided to candidates. Ensure all materials
        have clear identifiers, labels, or pre-defined categories that prevent ambiguity
        .]
    </MATERIALS_FOR_CANDIDATE>

    <MATERIALS_EXPLANATION_FOR_EVALUATOR>
    [Explain to the evaluator:
    - How the materials were created and what, if any, statistical patterns or other
        relationships exist
    - Cross-references or important conections between different materials (e.g., codes
        in a CSV that match details in text, or relationships between texts)
    - Any tricky elements or common pitfalls in the materials that may cause candidates
        to answer incorrectly
    - "Hidden" information that requires careful reading to identify]
    </MATERIALS_EXPLANATION_FOR_EVALUATOR>

    IMPORTANT: When designing the test, eliminate any opportunities for candidates to
        make arbitrary choices (like custom account codes, naming conventions, or
        classification systems) that would complicate evaluation. Either:
    - Provide pre-defined structures/codes that must be used, or
    - Design questions with objectively verifiable numerical/text answers that don't
        depend on the candidate's approach
```

```
- Make sure both start and end XML tags are present
```

Prompt 7: Fourth prompt in the exam generation pipeline: Submission instructions

```
Here is brief explanation of the exam's purpose and structure intended for the
    evaluator: <examoverview> {answer_overview}</examoverview>
Here are the instructions for the candidate: <instructions> {answer_instructions} </
    instructions>
Here are the materials provided to the candidate: <materials> {answer_materials} </
    materials>

## Your assignment
Based on the given information, specify exactly what format the candidate's answers
    must be in, including:
- Required JSON answer format with question IDs
- The exact format of answers (numbers, text, specific units, decimal places)
- Any supplementary files if necessary
- You should only specify format and/or code/conventions to use in answering, but you
     should not give the answers away
```

Prompt 8: Fifth prompt in the exam generation pipeline: Evaluation

```
Here is brief explanation of the exam's purpose and structure intended for the
    evaluator: <examoverview> {answer_overview}</examoverview>
Here are the instructions for the candidate: <instructions> {answer_instructions} </
    instructions>
Here are the materials provided to the candidate: <materials> {answer_materials} </
    materials>
Here are the submission requirements for the candidate: <submission_requirements> {
    answer_submission} </submission_requirements>

## Your assignment

Based on the given information create the following for the evaluator:
- Complete answer key in JSON format for automated checking
- Explanation of correct answers and how they were derived
- Passing criteria (e.g., minimum number of correct answers)
```

Prompt 9: Sixth prompt in the exam generation pipeline: Evaluation script

```
Here is brief explanation of the exam's purpose and structure intended for the
    evaluator: <examoverview> {answer_overview}</examoverview>
Here are the instructions for the candidate: <instructions> {answer_instructions} </
    instructions>
Here are the materials provided to the candidate: <materials> {answer_materials} </
    materials>
Here are the submission requirements for the candidate: <submission_requirements> {
    answer_submission} </submission_requirements>
Here is the information given to the evaluator: <evaluation_information> {
    answer_evaluation} </evaluation_information>

## Your assignment
Based on the given information, create a Python script named 'task_evaluation.py'
    that reads in the candidate submission and the answer key provided as arguments
    in the command line. The script should:
- Accept two arguments in the following order:
1. The **first argument** is the name of the candidate submission JSON file (e.g., `
    test_submission.json`).
2. The **second argument** is the name of the answer key JSON file (e.g., `answer_key
    .json`).
- Automatically score the test performance based on the provided files.
- Save the results as `test_results.json` in the same folder as the script.
- In addition to the detailed test results, `test_results.json` should include one
    variable `overall_score` with the percentage of points achieved by the candidate.
```

```
    The script should be runnable from the command line like this:
    ```bash
    python task_evaluation.py test_submission.json answer_key.json
```

B.3 Sanity checks

Prompt 10: Prompt to check wether the instructions falsely claim to include image materials

```
You are a system verifying if the provided instructions and/or materials falsely claim to
     include an image, but it is only a description.
    If the materials explicitly claim they have an image but only provide a textual
        description,
    and that image is crucial for at least one task, respond with "Y".
    In all other cases (including no mention of an image at all), respond with "N".
    Return only "Y" or "N".
```

Prompt 11: Prompt to check wether the instructions reference a fabricated website or news source

```

    You are a system verifying whether the instructions and/or materials provided
        reference a publicly available website or news source that appears to be
        fabricated.
    It is acceptable if the materials reference an internal document, company guidelines,
         accounting statements or well known public documents.
    However, if the materials claim to reference a real, publicly accessible website or
        piece of news (e.g., something you would expect to find only online)
    and that url or texts  appears to be made up, respond with "Y".
    In all other cases, respond with "N".
    Return only "Y" or "N".
```

Prompt 12: Prompt to check wether the exam overall made sense

```

You are a system verifying a remote, **practical** exam to assess a {state['occupation
    ']}'s ability to {state['task_description']}.

    CANDIDATE vs. EVALUATOR CONTEXT:
    - The candidate only sees: Instructions, Materials, Submission format.
    - The evaluator sees everything else (Overview, Evaluation info, Grading script, and
        the answer key).

    CHECKS TO PERFORM:
    1) Is this exam actually practical (testing real job tasks) rather than purely
        theoretical?
    2) Are the tasks realistic for a {state['occupation']} in the year 2025?
    3) Are the instructions, materials, and submission requirements unambiguous?
    4) Do the grading script and answer key correctly reflect the exam?
    - No scenario where a candidate can pass overall despite failing a critical part.
    - No scenario where a candidate who meets all requirements is incorrectly failed.
    - The answer key should score 100% on the grading script.

    HOW TO RESPOND:
    Return EXACTLY one JSON object. Here is the required structure (note the doubled
        braces to show literal braces in an f-string):

    {{
    "makes_sense": true,
    "explanation": "A concise explanation here. Also suggest potential weaknesses (e.g.
        key not scoring 100) or ambiguities in the exam."
    }}

    No additional text (e.g., disclaimers, markdown formatting) outside this JSON object.
        """

    user_message =
```
```

```
EXAM CONTENT:

{instructions}

{materials_candidate}

{submission}


ADDITONAL MATERIAL ONLY AVAILABLE TO THE EVALUATOR (NOT THE CANDIDATE):

Exam Overview:
{overview}

All materials:
{materials_all}

Evaluation information:
{evaluation}

Grading script
{grading}

Answer key (Evaluator-Only):
{answer_key}

(Note: The answer key is passed to the grading script as 'answer_key.json'.)
```

The Center on Regulation and Markets at Brookings provides independent, nonpartisan research on regulatory policy, applied broadly across microeconomic fields. It creates and promotes independent economic scholarship to inform regulatory policymaking, the regulatory process, and the efficient and equitable functioning of economic markets.

Questions about the research? Email communications@brookings.edu. Be sure to include the title of this paper in your inquiry.

The Brookings Institution | 1775 Massachusetts Ave. NW, Washington, DC 20036 | 202.797.6000