SYM-AM-24-072

Excerpt from the Proceedings of the Twenty-First Annual Acquisition Research Symposium

Acquisition Research: Creating Synergy for Informed Change

May 8–9, 2024

Published: April 30, 2024

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

The research presented in this report was supported by the Acquisition Research Program at the Naval Postgraduate School.

To request defense acquisition research, to become a research sponsor, or to print additional copies of reports, please contact any of the staff listed on the Acquisition Research Program website (www.acquisitionresearch.net).

ACQUISITION RESEARCH PROGRAM
DEPARTMENT OF DEFENSE MANAGEMENT
NAVAL POSTGRADUATE SCHOOL

Introducing SysEngBench: A Novel Benchmark for Assessing Large Language Models in Systems Engineering

Ryan Bell—is an 8-year experienced engineer in the defense industry. In his current role at Naval Information Warfare Center Atlantic (NIWC LANT), Bell provides modeling and simulation expertise to a variety of programs for the Navy and USMC. He specializes in simulating communication systems in complex environments and is an advocate for the use of digital engineering early in the systems engineering lifecycle. Bell earned a BS in electrical engineering from Clemson University and an MS in electrical engineering from Clemson University with a focus on electronics; he is currently pursuing his PhD in systems engineering at the Naval Postgraduate School. He is a South Carolina registered Professional Engineer (PE), published author, and teacher.

Ryan Longshore—is an 18-year veteran of both the defense and electric utility industries. In his current role at Naval Information Warfare Center Atlantic (NIWC LANT), Longshore leads a diverse team of engineers and scientists developing and integrating new technologies into command and operations centers. Longshore is heavily involved in the Navy's digital engineering transformation and leads multiple efforts in the model based systems engineering and model based engineering realms. Longshore earned a BS in electrical engineering from Clemson University and an MS in systems engineering from Southern Methodist University; he is currently pursuing his PhD in systems engineering from the Naval Postgraduate School. He is a South Carolina registered Professional Engineer (PE), an INCOSE Certified Systems Engineering Professional (CSEP), and has achieved the OMG SysML Model Builder Fundamental Certification.

Raymond Madachy, PhD—is a Professor in the Systems Engineering Department at the Naval Postgraduate School. His research interests include system and software cost modeling, affordability and tradespace analysis, modeling and simulation of systems and software engineering processes, integrating systems engineering and software engineering disciplines, and systems engineering tool environments. His research has been funded by diverse agencies across the DoD, National Security Agency, NASA, and several companies. He has developed widely used tools for systems and software cost estimation and is leading development of the open-source Systems Engineering Library (se-lib). He received the USC Center for Systems and Software Engineering Lifetime Achievement Award for "Innovative Development of a Wide Variety of Cost, Schedule and Quality Models and Simulations" in 2016. His books include Software Process Dynamics and What Every Engineer Should Know about Modeling and Simulation; he is the co-author of Software Cost Estimation with COCOMO II and Software Cost Estimation Metrics Manual for Defense Systems. He is writing Systems Engineering Principles for Software Engineers and What Every Engineer Should Know about Python.

Abstract

In the rapidly evolving field of artificial intelligence (AI), Large Language Models (LLMs) have demonstrated unprecedented capabilities in understanding and generating natural language. However, their proficiency in specialized domains, particularly in the complex and interdisciplinary field of systems engineering, remains less explored. This paper introduces SysEngBench, a novel benchmark specifically designed to evaluate LLMs in the context of systems engineering concepts and applications. SysEngBench will encompass a comprehensive set of tasks derived from core systems engineering processes, including requirements analysis, system architecture design, risk management, and stakeholder communication. By leveraging a diverse array of real-world and synthetically generated scenarios, SysEngBench aims to provide an assessment of LLMs' ability to interpret complex engineering problems and generate innovative solutions.

Our evaluation of leading LLMs using SysEngBench reveals significant insights into their current capabilities and limitations in systems engineering contexts. The findings suggest pathways for future research and development aimed at enhancing LLMs' utility in the systems engineering discipline. SysEngBench contributes to the understanding of AI's potential impact on systems engineering.

Keywords: Systems Engineering, Large Language Models (LLMs), Benchmark, SysEngBench, Performance Evaluation, Intelligent Decision Making

Introduction

The intersection of artificial intelligence (AI) and engineering presents a frontier with the potential to revolutionize how we approach complex engineering challenges. One field that focuses on architecting solutions for complex engineering challenges is systems engineering, an emerging engineering field that can capitalize on the widespread proliferation of AI to mature the field at a more rapid pace. In order to harness AI, an understanding must be established on how well Large Language Models (LLMs) perform within the field - an understanding that is not yet baselined for systems engineering. This paper seeks to target this knowledge gap with a targeted approach to assess the capabilities of LLMs within the domain of systems engineering.

This paper introduces SysEngBench, a pioneering benchmark designed to assess LLMs against a diverse set of concepts and applications encountered in systems engineering. The motivation behind SysEngBench stems from the recognition that there has been an evolution of benchmarks from common sense, to inference, to field specific. Evaluation of LLMs within field specific domains has already begun - from high school courses to medical exams to law exams (sources). As LLMs become more capable, more complex benchmarks must be made to continue to track progress. As benchmarks become more complex, field specific knowledge is necessary from practitioners and experts in the field. SysEngBench is the proposed benchmark for the systems engineering field and seeks to incorporate field practitioners and expert knowledge. The proposed framework is not all encompassing nor complete at this time of writing and seeks feedback from the community. More specifically, the objective of this paper is to provide the initial concept and framework of the benchmark to be molded.

Background and Related Work

Overview of Systems Engineering

Systems engineering stands at the convergence of various engineering disciplines, aimed at developing coherent and effective systems through a system lifecycle process. It involves methodologies and practices that ensure all aspects of a system's lifecycle, from conceptualization to decommissioning, are considered and optimized. This interdisciplinary approach addresses complexity by emphasizing robust planning, design, analysis, and management practices. Various methodologies are used within the systems engineering community, from the traditional systems engineering "vee" model, to the spiral model, to the waterfall model, and several others (Boehm, 1986).

Systems engineering spans across industries, including aerospace, automotive, software, and more, making systems engineering the glue to stitch together all of the other fields. In recent years, significant progress has been made with respect to Model Based Systems Engineering (MBSE) tools. These modeling and simulation tools help to understand the interlinking between industries and effectively manage the available trade space for any given system or system of systems modeled. MBSE tools sit at the intersection of modeling languages, structure, model based processes, and presentation frameworks.

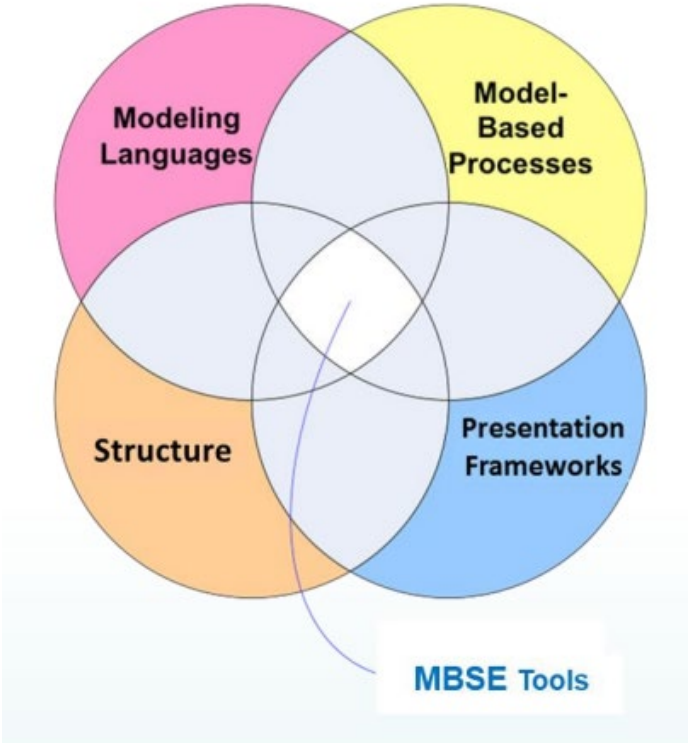

Figure 1. Model Based Systems Engineering Venn Diagram (Vaneman, 2020)

Front running tool sets include the likes of Cameo Enterprise Architect, Innoslate, among others. Most run on a UML backbone modified for systems engineering called SysML. Recent advancements have been made to SysML, known as SysMLv2, as an effort to democratize and open source systems engineering modeling. The architecture of SysMLv2 – which includes a textual format – will allow for a more fluid ability to train LLMs on models in the field.

The traditional systems engineering lifecycle is quite document-centric. In recent years, there has been a push to move towards model-centric management of the systems engineering lifecycle. Document-centric focuses on generating documents and those documents being the authoritative sources of truth for each of the milestones and associated efforts within the lifecycle, leading to an increasingly disaggregate pile of information – where sorting through this information to get a complete picture of how requirements and relationships within the system are represented also becomes increasingly complex. Due to the sheer amount of information and documentation, LLMs could significantly reduce the cognitive load and increase understanding of a systems current stature within the lifecycle, especially when aggregated into a single source of truth model (Defense Acquisition University [DAU], 2024; *SEBoK*, 2024).

Review of LLMs

Large Language Models (LLMs) such as GPT-4 have revolutionized the field of natural language processing (NLP) by demonstrating an impressive ability to understand and generate human-like text. These models are trained on vast amounts of text data, enabling them to grasp a wide range of topics, infer context, and produce coherent and contextually relevant responses. LLMs have been applied in numerous applications, from writing assistance and chatbots to more complex tasks like code generation and summarization.

Within the context of types of LLMs, there are different levels of accessibility, training sources, and varying levels of fidelity. For accessibility, there are open source models like Llama 2, Falcon, and Dolly as well as proprietary models like GPT-4, Claude, and Bard. Open source models are available in various repositories – one of the largest being HuggingFace. In general, proprietary models have been outperforming open source models, but the gap continues to close on the leaderboards. Every model is trained in a different set of data sources – some scrape GitHub for code, some scrape wikis and other openly available information or textbooks, and others are trained on private corpuses.

When it comes to fidelity, there are different preferences for fidelity based on the available hardware. A technique called quantization is common, where inference is ran on lower precision data types than the usual 32-bit (HuggingFace, 2024b). While this does result in lower fidelity, one can then run the model more easily on local hardware (Talamadupula, 2024). An example of a model released at varying data sizes is Llama 2, available in 7B, 13B, and 70B (*Llama 2: Open Foundation and Fine-Tuned Chat Models*, 2024).

To use models, different prompt can be used to change the output, a strategy described as prompt engineering. Prompts range from zero shot (no context provided and one try), to few shot (x number of refining attempts), and to Retrieval Augmented Generation (RAG). RAG pulls in relevant information from a large corpus of data at inference time when indexed properly. If prompt engineering does not give the desired answer, further fine-tuning of the model is required. Custom fine-tuning of LLMs on domain-specific datasets can significantly enhance their performance on specialized tasks.

Existing Benchmarks

The landscape of AI benchmarks has evolved over time, with early benchmarks focusing on foundational tasks such as work relationships and their semantic similarities to more recent, increasing complexity benchmarks such as College Medicine, Physics, Biology, Computer Science, Math, Electrical Engineering, among others (Hendrycks et al., 2021). Other non-technical outputs of LLMs are also being studied. The progression of increasing complexity is demonstrated in the table below, which shows the benchmark name, topic of the benchmark and the date the benchmark was initially released (*AI Fundamentals*, 2023). The list is not meant to be all encompassing or a review of literature, but rather a brief look at the evolution of benchmarks and their purpose over time.

Table 1. LLM Benchmarks over Time

Benchmark Name	Topic	Released	Type of Benchmark
WordNet	Word relationships and meanings, foundational dataset for semantic similarity and language understanding	1985	Natural Language Processing
MNIST	Handwritten digit recognition, foundational for image processing and computer vision	1998	Image Processing
BLEU	Language translation quality metric, foundational for evaluating machine translation systems	2002	Natural Language Processing
Enron Emails	Recognizing names, entities, and information extraction from natural email datasets	2004	Natural Language Processing
ImageNet	Large-scale image recognition and classification, pivotal in advancing deep learning in computer vision	2009	Image Processing

LAMBADA	Understanding context and reasoning in text, focusing on predicting sentence endings (Paperno et al., 2016)	2016	Natural Language Processing
SWAG	Common sense reasoning and predicting plausible sentence endings in a given context (Zellers et al., 2018)	2018	Natural Language Processing
GLUE	A collection of diverse NLU tasks like question answering and sentiment analysis to advance language understanding across various contexts.	2018	Natural Language Processing
SuperGLUE	A successor to GLUE with more challenging tasks, pushing the limits of NLU models with advanced reasoning and co-reference resolution.	2019	Natural Language Processing
HellaSWAG	An extension of SWAG for more challenging common sense reasoning scenarios (Zellers et al., 2019)	2019	Natural Language Processing
ARC	"ARC evaluates an AI's ability to tackle each task from scratch, using only the kind of prior knowledge about the world that humans naturally possess, known as core knowledge" (Clark et al., 2018; Lab42, 2024).	2019	Natural Language Processing
DROP	Reasoning over paragraphs, requires numerical reasoning and understanding of natural language (Dua et al., 2019)	2019	Natural Language Processing
Winogrande	A large-scale dataset of winograd schemas designed to improve commonsense reasoning in AI systems	2019	Natural Language Processing
XTREME	Cross-lingual understanding and translation across multiple languages, tests multilingual capabilities	2020	Natural Language Processing
MMLU	Measures professional and academic knowledge across various fields including College Medicine, Physics, Biology, Comp Sci, Math, Electrical Engineering, Professional Accounting, Psychology and worldly knowledge about Foreign Policy and Religions, among others (Hendrycks et al., 2021)	2021	Natural Language Processing
TruthfulQA	A question-answering dataset designed to evaluate a model's ability to produce truthful and factual answers.	2021	Natural Language Processing
GSM8K	Grade School Math 8K (GSM8K), a collection of math word problems aimed at evaluating numerical reasoning	2021	Natural Language Processing
BIG-Bench	Broad spectrum of tasks testing reasoning, common sense, professional knowledge, and language capabilities (*Google/BIG-Bench*, 2021/2024)	2022	Natural Language Processing

Performance of models across benchmarks are available in various locations, with the de-facto location being HuggingFace's (2024a) leaderboard. A list of other leaderboards is available on the site.

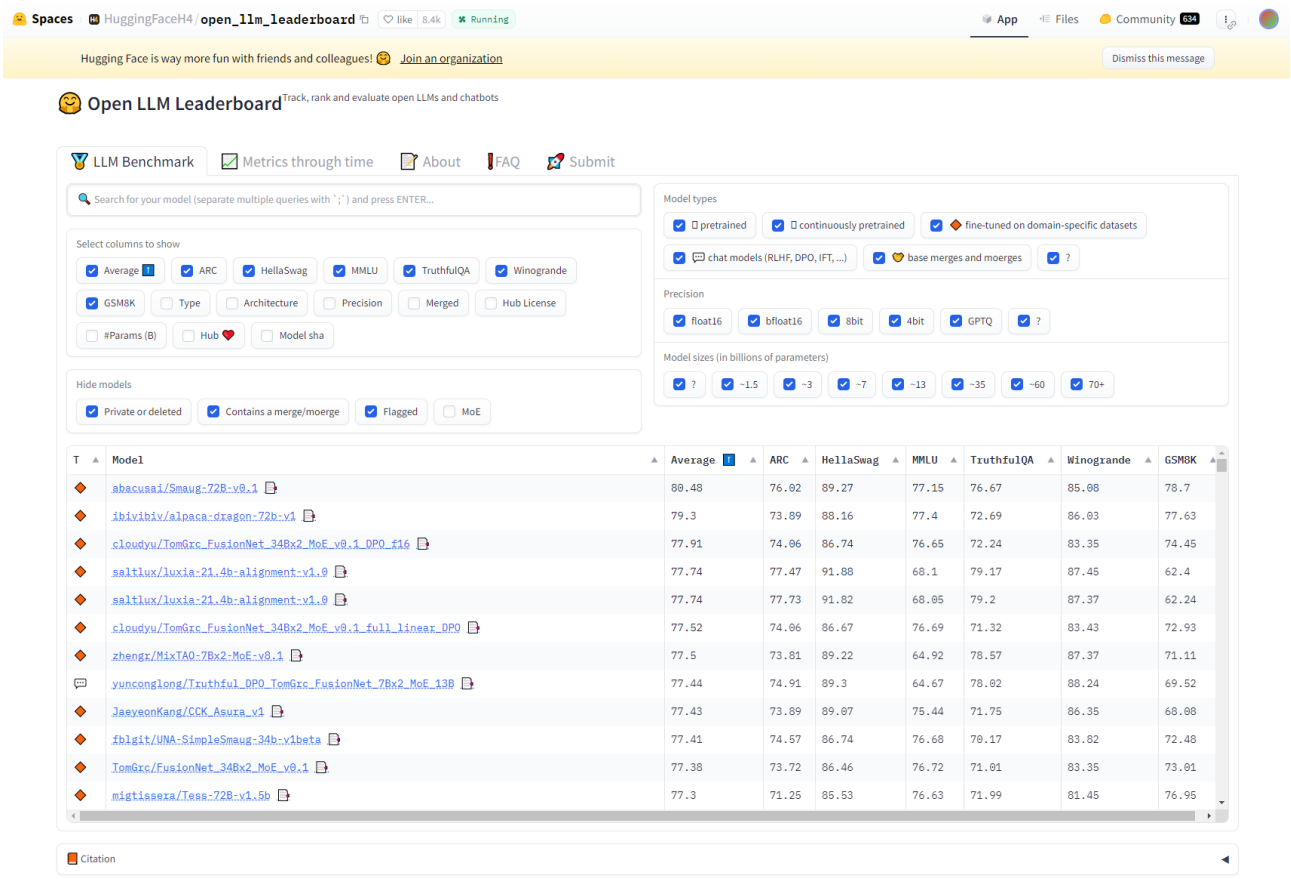

Figure 2. HuggingFace Leaderboards Screenshot

Benchmark Frameworks

Benchmarks range significantly when evaluating a domain specific field and level of complexity within that domain. Benchmarks can take a simple question and answer format, a multiple choice format, a fill in the blank format, an open ended response format, or various other methods. The more clear the answer has to be, the more clear the evaluation of a model with a given benchmark. Other more soft metrics are used for non-definitive answer scenarios to measure "correctness." This can range from measuring token counts, biases, tone, or otherwise.

Data Sources and Generation

Data sources for language models can vary widely. Some include professional documents only from journal articles and textbooks, while others also ingest blog posts and other sources, but one thing that remains common across all language models is that garbage in equals garbage out.

With regards to benchmark generation, datasets can be completely synthetic, semi-synthetic, or done completely by hand. Perhaps the worst quality assurance (QA) process is full synthetic, although for various types of data, this may be within acceptance criterion and the best method for creating data at scale (Lambert, 2023; Packt, 2024; *Synthetic Data*, 2024). For a domain specific application, semi-synthetic or by hand is recommended for the highest fidelity.

SysEngBench: Framework and Design

SysEngBench Framework

The framework selected for SysEngBench is a simple multiple choice question benchmark. The benchmark currently covers an introduction to systems engineering but will be expanded to sub-fields within systems engineering discussed in future work. The current fundamentals of systems engineering questions are questions that should be correctly answered by graduate level systems engineering students at least 1 year into their course work at the Naval Postgraduate School.

SysEngBench Categories

SysEngBench selects 10 topics to reflect the core processes of systems engineering. The 10 main areas and their descriptions can be found in Table 2 below.

Table 2. SysEngBench Topics

Area	Description
Requirements	Tasks that simulate the extraction, interpretation, and validation of system requirements from diverse sources, including stakeholder interviews and technical documents.
System Architecture and Design	Tasks that involve designing system architectures, considering aspects like modularity, scalability, resilience, and integration with existing systems.
Model-Based Systems Engineering (MBSE)	Tasks focusing on the application of modeling approaches to support system requirements, design, analysis, verification, and validation activities throughout the system lifecycle.
Cost Modeling	Tasks related to estimating, analyzing, and optimizing costs associated with the development and deployment of complex systems, taking into account factors such as materials, labor, and operational expenses.
System Capability/Suitability Engineering (-ilities)	Tasks aimed at evaluating and enhancing the overall performance and suitability of systems, including assessments of reliability, maintainability, and other critical 'ility' factors that affect system effectiveness and lifecycle cost.
Safety Engineering	Tasks involving the identification and mitigation of hazards, as well as the analysis of potential safety risks to minimize the likelihood and impact of accidents and failures.
Human Factors Engineering	Tasks that consider the interaction between humans and systems, aiming to optimize system performance via user interfaces, ergonomics, and usability studies.
System Integration and Development	Tasks focusing on the process of bringing together system components into a whole and ensuring that those components function together as intended, addressing challenges in integration and interoperability.
System Verification and Validation (V&V)	Tasks related to the confirmation that a system meets defined specifications and requirements (verification) and that it fulfills its intended purpose (validation), involving a combination of testing, analysis, and review techniques.
Risk Management	Tasks that require identifying potential risks, assessing their impact, and devising mitigation strategies, crucial for ensuring system reliability and safety.

The current iteration of the benchmark does not include all of the topic fields above since scope was currently limited to SE 3100 but is what will be strived for with future iterations, including refactoring the fundamentals tested into the proposed SysEngBench topics.

Table 3. Benchmark Question Distribution

Row Labels	Question Count	Question %
Fundamentals of SE	**116**	**100.00%**
SE Definitions	9	7.76%
Problem Definition and Stakeholders	11	9.48%
MBSE Overview	4	3.45%
Requirements	22	18.97%
Functional Analysis	11	9.48%
Value System Design	13	11.21%
Architecture	6	5.17%
Decision Making	10	8.62%
Risk	3	2.59%
System Integration, Qualification, Costs, Life Cycle Issues	27	23.28%
Grand Total	**116**	**100.00%**

SysEngBench Data Sources and Generation

The data sources used included lecture slides from SE 3100 at the Naval Postgraduate School. The syllabus for the class includes the following knowledge gained after taking the course:

- Define systems engineering, including its purpose and scope and the role of the systems engineer.
- Define systems architecting, including its purpose and scope and the role of the systems architect.
- Apply the fundamentals of a systems engineering process appropriately across a system's lifecycle.
- Elicit, elaborate and document system requirements based on user needs and operational objectives; translate them to technical requirements.
- Create a system value hierarchy reflective of stakeholder goals.
- Complete system functional analysis in support of requirements engineering using modeling tools such as IDEF0, FFBD and other techniques.
- Develop, evaluate and document alternative system architectures. A supplemental joint effort throughout the course will be to gain a common understanding of the applications of Systems Engineering in the Department of Defense (DoD).

The multiple choice questions were created with some AI assistance, but each was reviewed by a human systems engineer for correctness for a semi-synthetic dataset. More complex questions will investigate the LLMs ability to reason "within the gray" of systems engineering, particularly higher dimensional trade spaces where there are multiple configurations that would meet requirements.

Implementation and Benchmarking Process

Model Selection

A few common open source LLMs were selected for their availability and to show a range of performance. The models selected were Llama 2, Mistral, and Orca 2. All models used 8 bit quantization. The largest quantization available that would fit on a 32GB (or 64GB machine) was selected for each model.

Table 4. Models Used

Source	Model	Size	Quantization
TheBloke	Orca-2-7B-GGUF	7.16GB	8 bit
TheBloke	OpenHermes-2.5-Mistral-7B-GGUF	7.70GB	8 bit
TheBloke	Llama-2-7B-Chat-GGUF	7.16GB	8 bit

Benchmarking Procedure

The benchmarking process for SysEngBench is designed to be modular and replicable, as well as run locally or via cloud LLMs for future tests. To get some quick results, a simple evaluation method of querying for a response and parsing for a letter choice was implemented. To push for repeatability and scalability for future tests, lm-evaluation-harness will be implemented (*EleutherAI/Lm-Evaluation-Harness*, 2020/2024).

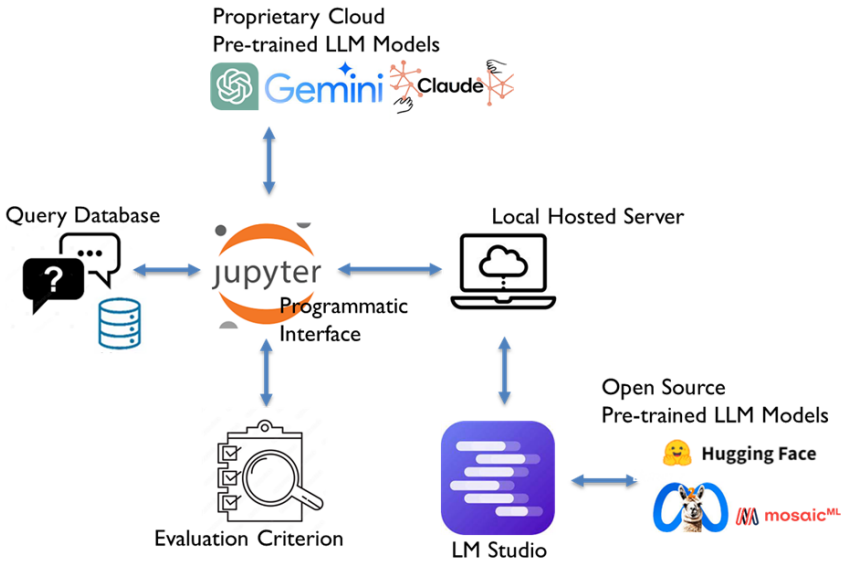

Figure 3. LLM Evaluation Framework

The code structure provided to the LLM of interest is below using LangChain. A zero shot method was used in the evaluation.

```python
import os
import json

results_dict = {}
file_output_name = 'llm_output.json'
file_path = os.path.join(folder_directory, file_output_name)
# Loop through each row in the DataFrame
instructions = "You are taking a multiple choice test. Only provide your letter answer, no explanation."
for index, row in df.iterrows():
    print("\nQuestion #" + str(row['Q#']) )
    prompt = f"Question: {row['Question']}\nChoices:\nA. {row['Choice A']}\nB. {row['Choice B']}\nC. {row['Choice C']}\nD.
    print(prompt)
    result = chat([SystemMessage(content=instructions), HumanMessage(content=prompt)])
    print(result.content)
    # Use the question number (or any unique identifier) as the key
    results_dict[row['Q#']] = result.content

    # Write the results_dict to a JSON file in the specified directory
    with open(file_path, 'w') as file:
        json.dump(results_dict, file)
```

Figure 4. Query and Response Code

Each language model's responses are to be scored against the correct answer key. The final performance of a model against the benchmark is to be represented as a percentage correct.

Results, Discussion, and Limitations

Results and Discussion

The implementation of SysEngBench across a range of LLMs, including both quantized models for local deployment yields insightful results into the capabilities and limitations of current AI technologies in the context of systems engineering. This section presents a summary of the findings, drawing comparisons between model performances and discussing the implications for the application of LLMs in systems engineering. The results for running the current state of the benchmark through open source LLMs is below in Figure 5.

			0.784482759	0.896551724	0.793103448
Row Labels	Question Count	Question %	20240320 LLaMA 2	20240320 Mistral	20240320 Orca 2
⊟ Fundamentals of SE	116	100.00%	91	104	92
SE Definitions	9	7.76%	8	9	9
Problem Definition and Stakeholders	11	9.48%	7	8	7
MBSE Overview	4	3.45%	3	3	3
Requirements	22	18.97%	17	22	15
Functional Analysis	11	9.48%	6	6	5
Value System Design	13	11.21%	12	12	13
Architecture	6	5.17%	3	5	4
Decision Making	10	8.62%	7	9	7
Risk	3	2.59%	2	3	3
System Integration, Qualification, Costs, Life Cycle Issues	27	23.28%	26	27	26
Grand Total	**116**	**100.00%**	**91**	**104**	**92**

Figure 5. Benchmark Evaluation Results

Out of the three models tested, the best performing model was Mistral at 89%, followed by Orca 2 at 79%, and Llama 2 at 78%. Perhaps the biggest delineating factor was performance of the models with Requirements questions, where Mistral was a clear leader with 22 correct out of 22, followed by Llama 2 with 17 and Orca with 15.

The worst performing topic for Llama 2 by percentage was architecture, for Mistral by percentage was functional analysis, and for Orca 2 by percentage was functional analysis as well. Should this trend continue, RAG or fine-tuning for functional analysis would be a potential knowledge gap solution, although not enough data points currently exist in the benchmark to statistically determine detrimental performance for the subtopics within systems engineering.

Challenges and Limitations

During the benchmarking process, a few challenges arose:

1. Very few LLM answers would have a letter selection followed by the choice verbiage and/or justification
2. Iterative refinement of the system message was required until the output was constant

Going forward, tighter integration with LangChain and lm-evaluation-harness should solve these issues.

The presence of variance by shifting which letter has the correct answer has been studied and is known to be present (Zheng et al., 2024). The variance for correct answer letter selection has not yet been investigated for this dataset.

The variance in different levels of quantization for different models was not tested. Open source versus proprietary was not yet tested, although the framework will allow for such an analysis in future work.

The level of complexity of questions within SysEngBench was also at a low complexity, focusing on high level concepts, and lacked a plethora of specific case studies.

Implications and Future Work

Implications for Systems Engineering

The SysEngBench benchmark has provided initial insight into capabilities and limitations of Large Language Models (LLMs) within the field of systems engineering. As the benchmark continues to be developed and LLMs progress over time, SysEngBench will allow for a reliable baseline for understanding model performance in systems engineering.

Eventual implications include enhanced efficiency and reduction of cognitive load required for tasks like documentation review, compliance checks, and stakeholder communications, enabling engineers to focus more on higher level aspects and navigating the available trade space of the complex system.

Future Directions and Related Work

The results of SysEngBench should be interpreted with consideration of its limitations, including the scope of tasks and the inherently complex nature of systems engineering characterized by the presence of multiple viable solutions.

Future iterations of the benchmark will incorporate a wider range of tasks, improved metrics for evaluating creative and integrative thinking, and direct comparisons with human performance to further refine our understanding of LLMs' potential in systems engineering. Various levels of complex questions, derived from a mix of real-world case studies, expertly crafted synthetic scenarios, and annotated datasets from academic and industry sources will be paramount.

A comprehensive list of future benchmark enhancements and research directions:

- **Complex Question Expansion**: To further challenge LLMs and accurately gauge their proficiency, SysEngBench will incorporate a broader array of complex questions and case studies that demand higher-order thinking, problem-solving, and the application of deep domain-specific knowledge. This expansion aims to push the boundaries of what LLMs can achieve within systems engineering.
- **Subfield Diversification**: Future iterations of the benchmark will expand upon the topic areas, such as safety engineering, logistics, sustainability, and human factors

engineering. This diversification will ensure that SysEngBench more fully represents the interdisciplinary nature of systems engineering and its varied applications across industries.

- **Evaluation by Practicing Systems Engineers**: Establish a comparative baseline and validate the benchmark's relevance; SysEngBench will be administered to practicing systems engineers. This initiative seeks to benchmark human performance against that of LLMs, offering invaluable insights into areas where AI can complement human expertise and identifying gaps where further AI development or human oversight is required.
- **Evaluation of Multiple Choice Question Bias within SysEngBench**: Evaluate the bias within multiple models across all topic areas to determine the variance of choosing correct answers. Leverage the techniques performed by Zheng et al. (2024).
- **Multimodal Input and Output**: Incorporate multimodal inputs (e.g., diagrams, charts, and technical drawings) and evaluating models' abilities to generate multimodal outputs could enhance the relevance and applicability of the benchmark to systems engineering practices.
- **Systems Engineering Domain Specific LLMs**: Investigate approaches to customize or specialize LLMs for specific domains within systems engineering via RAG or fine-tuning. Compare domain specific performance against the SysEngBench.
- **Enabling Round Table AI Discussions for an LLM SE Team:** Create a simulated team where multiple LLMs, each specialized in different aspects of systems engineering, can interact in a roundtable discussion format to tackle complex engineering challenges. The goal is to assess how well these AI models can collaborate, share insights, and come to a consensus or offer a range of solutions when confronted with multifaceted systems engineering problems.

Some of the future directions above include collaborations with others within the research group that are also working on the following topics:

- **Small Language Models for Domain Specific Knowledge:** Unlike their larger counterparts, these models aim to achieve deep expertise in narrow areas, potentially offering more precise and nuanced understanding and solutions. This approach could significantly enhance the quality of AI-generated recommendations and analyses in specialized fields, making these models invaluable tools for experts requiring detailed, domain-specific information.
- **Evaluation of LLMs with SysMLv2 Queries:** Evaluating LLMs' ability to understand and generate SysMLv2 queries represents a critical step towards integrating AI more deeply into systems engineering workflows. Current research investigates LLMs on their capacity to parse, reason about, and manipulate SysMLv2-based models, potentially automating or augmenting aspects of the systems modeling process (Longshore et al., in press). Success in this area could accelerate the model-based systems engineering (MBSE) process, making it more efficient and accessible.
- **Evaluation of LLMs for Modern Systems Engineering Cost Modeling with COSYSMO:** Constructive Systems Engineering Cost Model (COSYSMO) represents a cornerstone for estimating the costs associated with systems engineering projects. By evaluating LLMs on their ability to apply COSYSMO principles and methodologies, research can uncover AI's potential to revolutionize cost estimation in systems engineering in addition to accounting for AI productivity in novel cost factor modeling

(Madachy et al., in press). LLMs could assist in dynamically adjusting cost models based on real-time data and trends, offering a more agile and accurate approach to project management and budgeting in the field of systems engineering.

Conclusion

SysEngBench represents a significant advancement in evaluating the potential of Large Language Models within systems engineering, illuminating both the current capabilities and future possibilities of AI. By expanding the benchmark to encompass more intricate questions, a wider array of systems engineering subfields, and incorporating evaluations by practicing engineers, SysEngBench aims to bridge the gap between theoretical AI performance and practical engineering expertise. The evolving symbiotic relationship between AI development and systems engineering practice not only augments the capabilities of Large Language Models (LLMs) but also heralds a new era of engineering innovation characterized by collaborative partnerships between humans and AI. As we continue to explore the frontier of AI in systems engineering, the insights gained from SysEngBench will undoubtedly play a crucial role in shaping the future and maturing the discipline of systems engineering.

References

AI fundamentals: Benchmarks 101. (2023, April 7). https://open.spotify.com/episode/16vo3YLUTZI0nwAbrhjWYT

Boehm, B. (1986). *A spiral model of software development and enhancement*.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., & Tafjord, O. (2018). *Think you have solved question answering? Try ARC, the AI2 reasoning challenge* (arXiv:1803.05457; Version 1). arXiv. http://arxiv.org/abs/1803.05457

Defense Acquisition University. (2024). *Model-based systems engineering*. https://www.dau.edu/datl/b/model-based-systems-engineering

Dua, D., Wang, Y., Dasigi, P., Stanovsky, G., Singh, S., & Gardner, M. (2019). *DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs* (arXiv:1903.00161; Version 2). arXiv. http://arxiv.org/abs/1903.00161

EleutherAI/lm-evaluation-harness. (2024). [Python]. EleutherAI. https://github.com/EleutherAI/lm-evaluation-harness (Original work published 2020)

Google/BIG-bench. (2024). [Python]. Google. https://github.com/google/BIG-bench (Original work published 2021)

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2021). *Measuring massive multitask language understanding* (arXiv:2009.03300; Version 3). arXiv. http://arxiv.org/abs/2009.03300

HuggingFace. (2024a). *Open LLM leaderboard*. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

HuggingFace. (2024b). *Quantization*. https://huggingface.co/docs/optimum/en/concept_guides/quantization

Lab42. (2024). *About ARC*. https://lab42.global/arc/

Lambert, N. (2023, November 24). *Synthetic data: Anthropic's CAI, scaling, OpenAI's superalignment, tips, and open-source examples*. https://www.interconnects.ai/p/llm-synthetic-data

Llama 2: Open foundation and fine-tuned chat models. (2024). https://ai.meta.com/research/publications/llama-2-open-foundation-and-fine-tuned-chat-models/

Longshore, R., Madachy, R., & Bell, R. (in press). *Leveraging generative AI to create, modify, and query MBSE Models*. 21st Annual Acquisition Research Symposium.

Madachy, R., Bell, R., & Longshore, R. (in press). *Systems acquisition cost modeling initiative for AI assistance*. 21st Annual Acquisition Research Symposium.

Packt. (2024). *Generating synthetic data with LLMs*. https://www.packtpub.com/article-hub/generating-synthetic-data-with-llms

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q. N., Bernardi, R., Pezzelle, S. … Fernández, R. (2016). *The LAMBADA dataset: Word prediction requiring a broad discourse context* (arXiv:1606.06031). arXiv. https://doi.org/10.48550/arXiv.1606.06031

SEBoK. (2024). https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

Synthetic data: Save money, time and carbon with open source. (2024). https://huggingface.co/blog/synthetic-data-save-costs

Talamadupula, K. (2024, February 21). *A guide to quantization in LLMs*. Symbl.Ai. https://symbl.ai/developers/blog/a-guide-to-quantization-in-llms/

Vaneman (Director). (2020, March 31). *Webinar: Model-based systems engineering de-mystified*. INCOSE YouTube. https://www.youtube.com/watch?v=BPlphC88xR4

Zellers, R., Bisk, Y., Schwartz, R., & Choi, Y. (2018). *SWAG: A large-scale adversarial dataset for grounded commonsense inference* (arXiv:1808.05326; Version 1). arXiv. http://arxiv.org/abs/1808.05326

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., & Choi, Y. (2019). *HellaSwag: Can a machine really finish your sentence?* (arXiv:1905.07830; Version 1). arXiv. http://arxiv.org/abs/1905.07830

Zheng, C., Zhou, H., Meng, F., Zhou, J., & Huang, M. (2024). *Large language models are not robust multiple choice selectors*. The Twelfth International Conference on Learning Representations. https://openreview.net/forum?id=shr9PXz7T0

Acquisition Research Program
Department of Defense Management
Naval Postgraduate School
555 Dyer Road, Ingersoll Hall
Monterey, CA 93943

www.acquisitionresearch.net