A Design of Experiments Oracle LLM for Research-Grounded Combinatorial Testing

Logan Margabandu, Zizhao Chen*, W. Eric Wong, and Chih-Wei Hsu
The University of Texas at Dallas, Richardson, Texas, United States
lmm220012@utdallas.edu, zxc190007@utdallas.edu, ewong@utdallas.edu, cxh210019@utdallas.edu
*corresponding author

Abstract—As artificial intelligence moves across all scientific and engineering disciplines, a big challenge remains: expert domain knowledge, especially in specialized areas like Design-of-Experiments (DoE), is hard to interpret or inaccessible to non-experts and thus hinders broader application and innovation. As large language models (LLMS) continue to improve, they offer a practical way to help close the knowledge gap and make expert-level information more accessible to a wider audience. To address this need, we designed and developed a new solution: a Design-of-Experiments Oracle (DOE), a conversational LLM. This paper presents this AI-powered system that changes how complex DoE principles are understood and used. DOE uses state-of-the-art techniques including a Llama 3.2 model inside a Retrieval-Augmented Fine-Tuned Transformer (RAFT) architecture. Its knowledge base is informed by and enables semantic retrieval from combinatorial testing research. This is a conversational expert system. With this system, we want to enable engineers and researchers to apply complex DoE principles, give them interactive, research-based guidance. Ultimately, this is to contribute to the efforts to speed up innovation, improve system reliability and make advanced engineering methods more accessible in the digital age.

Keywords—Design-of-Experiments (DoE); Oracle LLM; Conversational AI; Large Language Models; Retrieval Augmented Generation (RAG); Combinatorial Testing;

1. INTRODUCTION

Design-of-Experiments (DoE) offers a practical set of statistical methods that are widely used to improve processes and develop products in many areas of science and engineering [1]. As modern engineering and scientific problems escalate in complexity, it becomes increasingly challenging for non-expert practitioners to understand and implement these intricate DoE concepts [2]. This has evolved into a major bottleneck, limiting innovation and efficiency across many sectors—a difficulty further underlined by research on applying systematic testing techniques such as Combinatorial Testing (CT) in real-world contexts [3]. Consequently, extensive study has aimed to create tools and methods to close this knowledge gap [2], yet a truly intuitive, interactive, and research-grounded expert assistant has remained an elusive goal. This document discusses the design, development, and implementation of the Design-of-Experiments Oracle LLM (DOE). This work aims to combine developments in large language models (LLMs) with basic DoE research, especially using CT's cost-effectiveness in producing high-quality test cases [4], thereby creating a novel system—DOE—that offers an accessible and robust solution for navigating the complexities of experimental design.

The foundational yet often intricate nature of certain DoE techniques, such as those in combinatorial testing, highlights the need for such expert systems. For example, empirical studies applying combinatorial testing in industrial environments have established highly effective strategies for identifying critical system defects with significantly reduced testing effort [5]. This body of work offers a very effective framework for guaranteeing system and software dependability. However, the practical use of these sophisticated n-way interaction testing methods, choice of suitable covering arrays, and interpretation of resulting data calls for a nuanced understanding that typically resides only with seasoned DoE specialists, a common challenge observed when implementing CT in industry [3]. Many engineering teams may not fully utilize these effective approaches because of their dependence on scarce knowledge, which could result in suboptimal testing coverage, missed critical defects, or ineffective resource allocation [2,3]. The challenge, therefore, is not merely the existence of potent methodologies, but their effective translation and accessibility to the broader engineering community that stands to benefit most. Our DOE is conceived as a direct response to this translation and accessibility imperative.

Recent advances in large language models (LLMs) have opened up new opportunities to improve how expert knowledge is shared and used [6,7]. Often based on sophisticated architectures like Transformers [8], these models have demonstrated remarkable capabilities in natural language understanding, generation, and complex reasoning. This has led to transformative improvements in areas such as automated question-answering, knowledge synthesis, and even code generation. However, general-purpose LLMs, despite their breadth, inherently lack the deep, verifiable, and specialized expertise required for nuanced scientific and engineering domains like Design-of-Experiments without targeted augmentation. The risk of relying on ungrounded

2693-9371/25/$31.00 ©2025 IEEE
DOI 10.1109/QRS-C65679.2025.00021

LLMs for critical tasks underscores the necessity for systems that are not just intelligent but also deeply anchored in authoritative domain-specific knowledge. This opens a crucial path for innovation: the development of specialized LLM-based systems that act as expert "oracles," capable of guiding users through complex decision-making processes. Imagine an engineer facing the daunting task of designing an optimal test suite for a safety-critical system; sifting through dense academic papers or navigating complex statistical software for DoE guidance can be prohibitively time-consuming and error-prone, as engineers often find existing technical research on methods like CT difficult to follow [3]. DOE, as presented in this paper, is engineered to alleviate precisely these burdens, offering an interactive, conversational interface to the rich, specialized knowledge of combinatorial testing research.

The remainder of this paper is organized as follows: Section 2 reviews related works in Large Language Models, Combinatorial Testing, and the broader field of Design-of-Experiments. Section 3 details the methodology employed for developing the application DOE, including knowledge base curation, question-answer pair generation, and model architecture. Section 4 presents the results of our system evaluation and its implications. Finally, Section 5 concludes the paper.

2. Related Work/Study

This section introduces the core areas that shaped the development of the Design-of-Experiments Oracle LLM (DOE). It begins by covering recent progress in large language models (LLMs) and the underlying architectures that support them. Next, it examines Combinatorial Testing (CT), emphasizing key research findings and the challenges of applying CT in practice. Finally, it places our work within the broader framework of Design-of-Experiments (DoE).

2.1 Large Language Models (LLMs)

The rapid rise of large language models (LLMs) has significantly influenced the field of artificial intelligence, especially in natural language processing (NLP). Built with billions of parameters and trained on massive text datasets, these models have shown strong capabilities in understanding, generating, and reasoning with human language. A key component behind their success is the Transformer architecture, introduced in prior research, which uses attention mechanisms to process input sequences in parallel—addressing the limitations of earlier recurrent models [8]. This design has allowed LLMs to perform well across a range of tasks, such as answering complex questions, producing coherent text, and summarizing information effectively.

A critical advancement for enhancing the factual grounding and domain-specificity of LLMs is Retrieval Augmented Generation (RAG) [9]. RAG frameworks address inherent limitations of LLMs, such as knowledge cut-offs and potential for hallucination, by dynamically retrieving relevant information from external knowledge sources before generating a response. This approach allows LLMs to access up-to-date or specialized information, making their outputs more accurate, verifiable, and contextually appropriate. The capabilities of LLMs, particularly when augmented with RAG techniques, provide a robust foundation for developing interactive expert systems, such as our DOE, which aims to provide specialized, research-grounded guidance. Our system utilizes the LLaMA 3.2 model, an example of current state-of-the-art LLMs, integrated within a Retrieval-Augmented Fine-Tuned Transformer (RAFT) architecture to deliver its expert capabilities.

2.2 Combinatorial Testing (CT)

Combinatorial Testing (CT) is a black-box software testing technique designed to efficiently detect faults caused by interactions among system parameters or configurations. Instead of exhaustively testing all possible combinations, which is often infeasible, CT aims to cover all t-way interactions (where t is typically a small integer, e.g., 2 to 6) using a significantly smaller set of test cases. This approach is predicated on the empirical observation that most software faults are triggered by interactions involving only a few parameters.

The practical value and limitations of combinatorial testing (CT) in real-world settings have been well studied by Dr. Eric Wong and other researchers. They found that CT was applied in industrial environments and found to detect more bugs while reducing the time needed for test case design compared to traditional function coverage-based methods [3]. Their findings underscore CT’s potential as a cost-effective approach for testing a variety of software systems, including embedded and operating systems, while also acknowledging challenges such as managing parameter configurations and the need for robust tool support.

Further empirical validation in the study on five industrial systems with real faults reinforced CT's effectiveness, especially in detecting multi-factor faults [3]. This study emphasized the critical importance of detailed requirement documents for constructing high-quality input space models (ISMs)—a crucial step for CT's success. It also identified practical challenges, including the potential for test execution and output verification to mask faults if not managed carefully, and the difficulty in applying higher-strength CT to complex systems without robust automation and efficient test suite management.

Moreover, the capability of CT to enhance fault detection strength and achieve rigorous coverage criteria, such as Modified Condition/Decision Coverage (MC/DC) required for safety-critical systems, was also explored in this study [4]. The research indicated that CT can be effectively used to generate test cases that achieve high MC/DC, further underscoring its value in producing high-quality, reliable software.

Despite its demonstrated benefits, the application of CT can be challenging for practitioners due to the complexities in identifying relevant parameters and their values, constructing effective ISMs, selecting appropriate interaction strengths, and interpreting the combinatorial test suites. The DoE-Oracle aims to mitigate these challenges by providing an interactive, conversational interface that guides users through the principles of CT, leveraging the foundational research of Dr. Eric Wong and other researchers to make this powerful testing methodology more accessible.

2.3 Design-of-Experiments (DoE)

Design-of-Experiments (DoE) refers to a set of structured statistical methods used to analyze and improve processes by systematically varying input factors and observing their impact on outcomes. It plays a central role in scientific and engineering research by helping identify key factor interactions, streamline experimentation, and develop more reliable products and systems. Core principles of DoE include randomization (to reduce bias), replication (to measure experimental error), and blocking (to control for known sources of variability) [1]. Common approaches include factorial and fractional factorial designs, response surface methodology, and Taguchi methods. In the context of software testing, combinatorial testing can be seen as a focused application of these DoE principles.

While powerful, the broader application of DoE techniques often faces hurdles. These include the inherent statistical complexity, the extensive array of available experimental designs (making selection difficult for non-statisticians), and the challenges associated with analyzing and interpreting the experimental results [2]. Although various statistical software packages (e.g., JMP, Minitab, R) provide tools for designing and analyzing experiments, they typically require a significant level of statistical expertise and may not offer intuitive guidance for users unfamiliar with DoE concepts.

Our DoE-Oracle (DOE), which initially focuses on making Combinatorial Testing more accessible, targets a specific area within the broader field of Design-of-Experiments. At its core, the system uses an LLM-driven conversational interface to present complex experimental design concepts in a more approachable way. By offering guidance grounded in research through an intuitive format, it reduces the learning curve associated with traditional tools and technical literature, supporting wider adoption of sound experimental design practices.

3. METHODOLOGY

The development of the Design-of-Experiments Oracle LLM (DOE) followed a structured, research-informed process designed to build a reliable conversational assistant for combinatorial testing. The methodology was divided into several key stages: selecting and analyzing foundational research, generating domain-specific question-answer pairs, incorporating distractors to improve retrieval precision, and integrating all components into a retrieval-augmented language model. These stages were shaped by best practices in domain adaptation for language models and grounded in empirical research [8,10].

3.1 Establishing Foundational Research and Knowledge Base Curation

The foundation of the DOE system is a specialized knowledge base developed from a curated selection of seminal research papers in Combinatorial Testing (CT), primarily authored by Dr. Eric Wong and his collaborators. These papers were selected for their influence in the field and their focus on both theoretical foundations and real-world applications of CT in industrial settings [3].

To clarify the technical steps involved in preparing the model for domain-specific adaptation, Figure 1 illustrates the pretraining process from the engineer's perspective. This includes uploading the dataset, initiating the training procedure, updating the model's internal knowledge representation, and managing potential failures during training

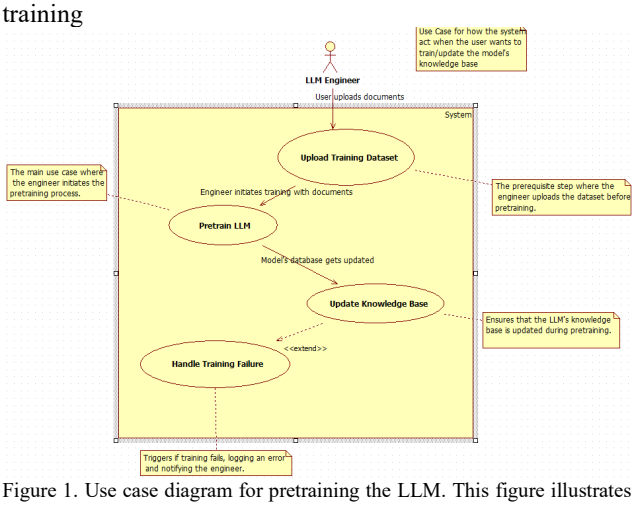

Figure 1. Use case diagram for pretraining the LLM. This figure illustrates the workflow involved in updating the model's knowledge base, including dataset upload, initiation of the training process, database update, and error handling during training failure.

Once the source materials were selected, each paper underwent multiple readings. During this review, detailed annotations were made to identify core definitions, important findings, methodological patterns, and areas that could potentially confuse or challenge end users. This thematic analysis approach was used to ensure a thorough understanding of the material and to guide the development of high-quality, domain-specific guidance [11]. The resulting understanding formed the basis of the Q&A dataset that powers the DOE system's retrieval and generation functions.

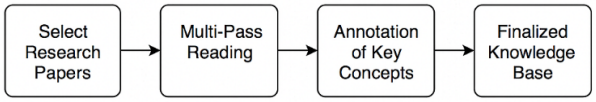

Figure 2. Knowledge Base Curation Pipeline. This diagram illustrates the sequential process used to construct the DOE Oracle's domain-specific knowledge base, including paper selection, multi-pass reading, annotation of key concepts, and thematic synthesis leading to a finalized corpus for Q&A generation.

3.2 Domain-Specific Question-Answer Pair Generation

A central component of the DOE Oracle's training data was a carefully constructed set of question-answer (Q&A) pairs derived directly from the curated combinatorial testing research papers. This multi-stage process was designed to ensure factual accuracy, depth, and relevance across a range of user queries.

The process began with an initial round of close reading and annotation, from which a comprehensive list of candidate questions was iteratively developed. These questions were designed to span multiple levels of cognitive engagement—from factual recall (e.g., "What is t-way combinatorial testing?"), to methodological reasoning (e.g., "How is an Input Space Model constructed in CT?"), to interpretation and comparison (e.g., "How does CT perform relative to functional testing in industrial settings?"). Each question was tailored to reflect realistic information needs that practitioners might face when working with combinatorial testing tools or principles.

After formulating the questions, the research papers were re-analyzed with the sole purpose of locating authoritative content to construct answers. A strict constraint was applied: every answer had to be explicitly supported by, and synthesized from, the primary texts. No external assumptions or interpolations were permitted, ensuring that the DOE system remains grounded in verified domain knowledge [12].

Each Q&A pair then underwent multiple rounds of quality assurance. This included verifying conceptual accuracy, ensuring clarity and conciseness, and cross-referencing with the source material to eliminate potential misinterpretations. The final dataset was validated using a golden test set—an expertly curated benchmark provided by a domain expert familiar with the original research [10]. This benchmark helped ensure the generated responses reflected the intent, scope, and technical accuracy of the foundational research.
To provide a clear overview of this development process, Figure 3 illustrates the full pipeline used to generate and validate the question-answer pairs used in training the DOE Oracle.

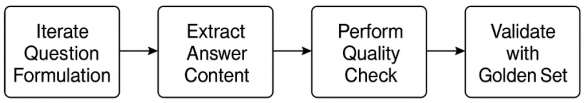

Figure 3. Question-Answer Pair Creation Pipeline. This diagram summarizes the multi-stage process used to generate high-quality Q&A pairs for training the DOE system, including formulation, answer extraction, review, and validation.

This structured approach enabled the system to deliver responses that are not only accurate and informative but also tightly aligned with expert-authored combinatorial testing research.

3.3 Incorporating Distractors for Enhanced Relevance Discrimination

To improve the precision and reliability of the DOE Oracle, the system was designed to identify and down-rank misleading or irrelevant content during retrieval. A key component of this strategy involved the incorporation of *distractors*—carefully crafted instances of near-relevant or subtly misleading content used to test and refine the model's ability to distinguish between useful and non-useful information.

These distractors were developed through a structured process. Some were extracted from the combinatorial testing research but only tangentially related to specific questions. Others were factually incorrect statements that mimicked the tone of valid claims, or content drawn from general Design-of-Experiments or software testing topics that lacked direct applicability to the targeted combinatorial queries. By introducing these distractors into the training and evaluation pipeline, the system was exposed to examples where context appeared relevant on the surface but failed to provide the necessary substance.

In the retrieval phase, distractors served as negative examples, enabling the retriever to learn which passages should be ignored despite lexical similarity to the query. This form of contrastive learning has been shown to significantly enhance retrieval accuracy in open-domain question answering systems by emphasizing semantic precision over superficial matches [13].

Figure 4 illustrates this mechanism by showing how distractor passages are incorporated into the training pipeline to improve relevance discrimination.

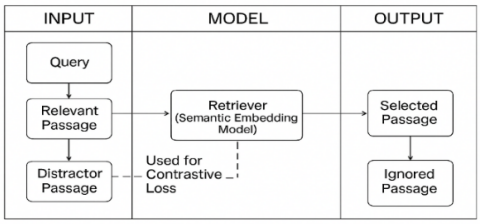

Figure 4. Use of distractors for enhanced relevance discrimination. This diagram illustrates how the retriever model leverages both relevant passages and distractor passages during training. Distractors serve as negative examples contributing to contrastive loss, enabling the model to prioritize contextually appropriate passages and down-rank irrelevant content.

In the generation stage, distractors challenged the model to extract correct answers only from genuinely relevant passages, discouraging it from synthesizing information from unrelated or misleading context.

Overall, this methodology improved the DOE Oracle's ability to produce context-aware responses while minimizing the influence of noise. The system became more discerning, robust, and less prone to producing answers based on keyword proximity rather than true semantic alignment [14].

3.4 Model Architecture and Data Utilization

The DOE Oracle system is built upon a Retrieval-Augmented Fine-Tuned Transformer (RAFT) architecture, integrating the LLaMA 3.2 model as its foundational large language model. This framework combines domain-specific fine-tuning with retrieval mechanisms to support high-fidelity, context-aware question answering.

The knowledge base was assembled by chunking full-text research papers in combinatorial testing, which were then embedded and indexed using the FAISS (Facebook AI Similarity Search) framework. This enables efficient semantic retrieval of relevant passages in response to user queries, allowing the model to ground its answers in authoritative content [15].

During inference, when a user submits a question, semantically similar chunks are retrieved from the FAISS index and passed to the LLM. This retrieval is informed by the Q&A pairs, which serve both as training data and as representative prompts for structuring valid queries and answers. These curated examples guide the model in understanding how to synthesize accurate and relevant responses from the retrieved context [16].

The fine-tuning component was facilitated using PyTorch and TensorFlow, which supported early stage experimentation and architectural adjustments. In particular, Apple's MLX framework was employed to accelerate training on Apple Silicon hardware, ensuring compatibility and performance for local development and iterative prototyping [17].

The RAFT system also uses distractors—plausible but incorrect content—as negative examples during training. These play a key role in contrastive learning by helping the retriever distinguish relevant from irrelevant information, which in turn improves the quality of generated responses by encouraging the model to rely only on accurate, context-specific input [18].

This modular design, blending retrieval and generation through a fine-tuned transformer pipeline, allows the DOE Oracle to offer expert-level guidance grounded in verifiable research, optimizing both precision and contextual awareness.

Figure 5 illustrates the overall architecture of the DOE Oracle system, including the integration of the retriever, fine-tuned generator, distractors, and evaluation components.

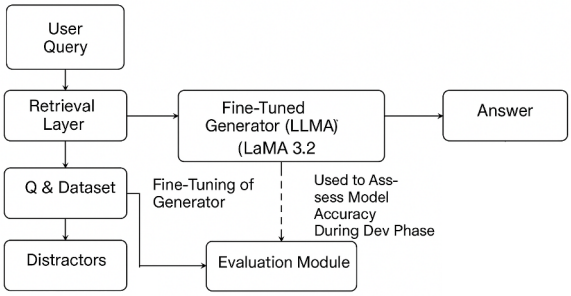

Figure 5. RAFT Model Architecture. This diagram shows the high-level structure of the DOE Oracle's RAFT system, including the curated knowledge base, FAISS index, fine-tuned LLM (LLaMA 3.2), Q&A dataset, and distractors used for contrastive training and evaluation. The model retrieves semantically relevant passages to synthesize domain-specific answers with enhanced accuracy and contextual alignment.

This architectural design emphasizes modularity and interpretability. By combining a semantically indexed knowledge base with a fine-tuned LLM and contrastively trained retriever, the DOE Oracle can produce highly relevant, evidence-backed responses to complex queries. The use of distractors ensures robustness, while the Q&A-driven fine-tuning aligns the system with expert reasoning patterns. This integration allows the system to scale effectively to other domains with minimal structural change.

3.5 System Implementation

The practical realization of this methodology involved leveraging several key technologies. LangChain was utilized to orchestrate the components of the RAG pipeline, including

the interaction between the LLM, the FAISS vector store, and the prompting strategies. The backend was developed using Flask, complemented by a React-based frontend for user interaction. Docker was employed for containerization, facilitating deployment on the Google Cloud Platform (GCP).

3.6 Citation Injection for Source Traceability

To enhance response trustworthiness, the system was designed to inject citation metadata during the retrieval phase. Each document chunk stored in FAISS was tagged with its originating paper title, author, and year. When the top-k relevant chunks were retrieved, these tags were preserved and embedded directly into the context prompt sent to the language model. This allowed the Oracle to reference specific sources during generation (e.g., "[Wong et al., 2016]"), enabling traceability of each fact presented. This strategy aligns with best practices in retrieval-augmented generation, where grounded responses are crucial for mitigating hallucinations [8].

4. RESULTS

The Design-of-Experiments (DoE) Oracle LLM met its primary objective of enabling interactive, domain-specific guidance on combinatorial testing through a retrieval-augmented and fine-tuned language model pipeline. The system was developed using a RAFT (Retrieval-Augmented Fine-Tuned Transformer) approach, which combined vector-based retrieval (FAISS) with a locally hosted LLaMA 3.2 model via Ollama. This enabled the Oracle to produce responses that were closely aligned with the academic work of Dr. Eric Wong on combinatorial testing.

Qualitative testing demonstrated that the Oracle reliably answered complex queries such as "What is t-way coverage?" and "How does IPOG compare to other test generation algorithms?" with contextually accurate and research-supported responses. These outputs were validated against a golden test set curated by a domain expert, ensuring fidelity to the source material.

The system was evaluated using both black-box and white-box testing techniques to ensure functional correctness, output reliability, and internal consistency [19]. Black-box testing served as the primary validation method, emphasizing external behavior based on user queries and expected responses. As demonstrated in Table I, test cases such as TC002 and TC003 validated the system's ability to provide accurate, grounded explanations of core combinatorial testing concepts, as well as its ability to include correct citations from authoritative sources.

White-box testing was used to assess the internal processing pipeline, including document ingestion, chunking, embedding generation, retrieval accuracy, and prompt construction. These tests ensured that each component behaved as intended and contributed to a coherent and context-aware response generation process. For example, TC009 and TC010 verified chunk segmentation and embedding vector dimensions, while TC011 and TC014 confirmed the accuracy of semantic retrieval and citation metadata injection. This dual testing strategy ensured that both user-facing behavior and internal system logic were thoroughly validated.

Table I. Functional test cases used for black-box evaluation of the DoE Oracle LLM

Test Case ID	Test Description	Pre-conditions	Test Steps	Expected Results
TC001	Verify user can access the DOE Oracle interface	User is on the main app; the system is online.	1. Navigate to the DOE Oracle page 2. Click 'Start Chat' or equivalent button	Chat interface loads and is ready for input.
TC002	Test DOE Oracle response to a general CT question	DOE Oracle is active; user is at the chat interface	1. Type a question such as 'What is combinatorial testing?' 2. Press enter or click send	DOE Oracle replies with a clear and accurate definition of combinatorial testing.
TC003	Test DOE Oracle's ability to cite relevant research	DOE Oracle is active and has access to Dr. Wong's papers	1. Ask a specific question (e.g., 'How does 2-way CT compare to traditional testing?')	DOE Oracle includes accurate information from a known source with inline citation or a clear reference.
TC004	Test DOE Oracle fallback response to out-of-scope question	DOE Oracle is active	1. Ask an unrelated question (e.g., 'What's the capital of Canada?')	DOE Oracle returns a fallback or polite clarification message stating it is only trained on CT research.
TC005	Measure DOE Oracle response latency	DOE Oracle is active and connected	1. Submit a known question 2. Start timer on submission 3. Stop timer on reply	DOE Oracle responds within an acceptable latency threshold (e.g., under 13 seconds).
TC006	Verify DOE Oracle handles multi-turn follow-up questions	DOE Oracle is active	1. Ask: 'What is IPOG?' 2. Follow up with: 'How does it compare to other algorithms?'	DOE Oracle maintains context and responds appropriately to the follow-up.
TC007	Confirm that DOE Oracle does not hallucinate or fabricate unknown information	DOE Oracle is active	1. Ask a borderline or misleading question (e.g., "How does CT reduce time complexity of QuickSort?")	Chatbot responds with either a clarification or rejection of unsupported claims.
TC008	Verify DOE Oracle provides references from specific uploaded papers	DOE Oracle has been trained or fine-tuned on specific CT papers	1. Ask: 'What does the 2016 industrial case study written by Eric Wong and collaborators say about 2-way CT?'	DOE Oracle extracts context from the correct paper and provides an answer aligned with the content.

The testing strategy emphasized three key evaluation criteria: user satisfaction (as observed during interactive sessions), scenario coverage (including fallback behavior, latency, and multi-turn follow-up), and maintainability (enabled by a modular backend architecture). This structured approach contributed to a robust, reliable user experience and ensured traceability between requirements, test cases, and system behavior [20].

The system's RAFT architecture preserved context effectively across multi-turn interactions, allowing for follow-up questions without repetition or hallucination. When presented with questions outside the domain, the Oracle appropriately issued clarification responses or refrained from answering—an expected behavior in retrieval-augmented frameworks designed to reduce unsupported generations [21].

Functional correctness was confirmed through black-box test cases aligned with user roles and key use cases. Test inputs

were mapped to anticipated outcomes and verified through manual review. The system also underwent white-box evaluations to ensure that chunking, embedding, and top-K retrieval behaved as expected. Embedding dimensions and retrieval accuracy were manually cross-verified to maintain semantic integrity.

A critical feature of the Oracle LLM is its ability to retrieve and present source-grounded responses. The system injects inline citations into retrieved chunks before generation, allowing users to trace answers back to original sources—a strategy that enhances trust and reduces hallucinations [22].

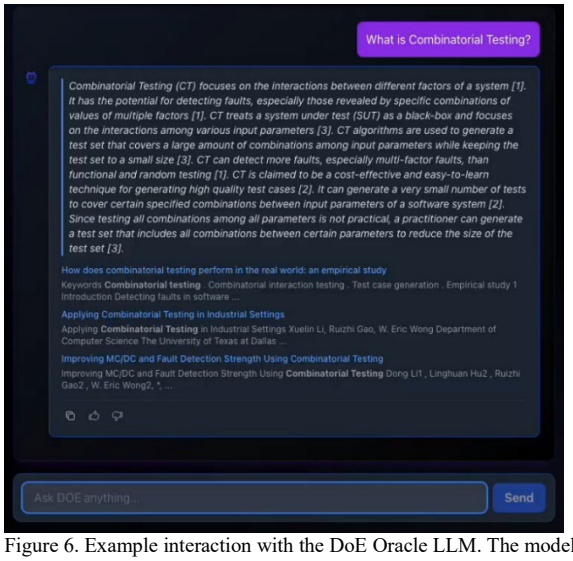

Figure 6. Example interaction with the DoE Oracle LLM. The model responds to a domain-specific query with an explanation of combinatorial testing, including inline citations ([1], [2], [3]) linked to source documents. This citation mechanism enhances trust and reduces hallucinations by grounding responses in credible research.

Two deployment pathways were validated: a cloud-hosted web interface and a locally installable MSI package. Both instances maintained consistent functionality, demonstrating the system's portability and reliability—critical attributes in AI-driven developer tools [22].

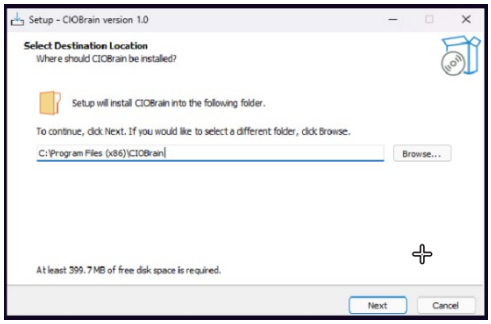

Figure 7. MSI installer interface for local deployment of the Oracle LLM. Users can select an installation path, and the setup process includes all required dependencies for offline use.

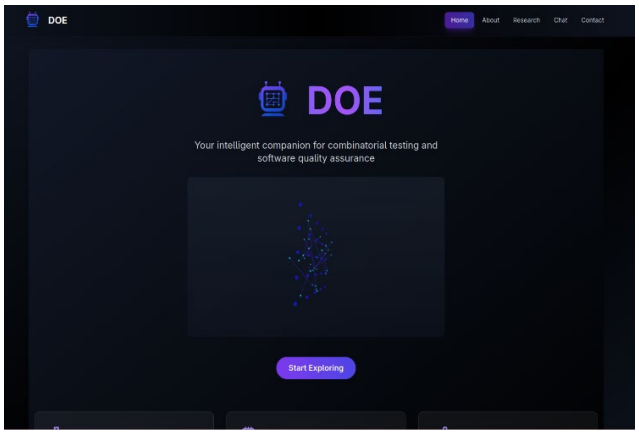

Figure 8. Screenshot of the DoE Oracle LLM web interface. The front end, built with React and Tailwind CSS, allows users to submit domain-specific queries and view citation-backed responses.

Overall, the Oracle LLM's results confirm that RAFT-based integration can deliver domain-grounded and context-aware responses in a scalable and user-friendly format. The modular design supports future extensions to other software engineering domains while maintaining rigorous control over response validity and traceability.

5. CONCLUSION & FUTURE DIRECTIONS

The Design-of-Experiments (DoE) Oracle LLM was created to provide targeted, research-informed guidance for combinatorial testing. Built on a retrieval-augmented and fine-tuned transformer model, the system integrates a curated research base with FAISS-based semantic retrieval and local LLM inference. This architecture enables the Oracle to respond to technical queries with contextually accurate and verifiable information derived from foundational research in the field.

Evaluation of the system demonstrated that it could reliably interpret and answer domain-specific questions related to testing strategies, such as t-way coverage, masking effects, and test generation algorithms. These capabilities were verified through black-box testing focused on user-facing behavior, and white-box testing that examined the internal processing pipeline. Components such as chunking, embedding, retrieval, and citation injection were each validated to ensure the system functioned predictably and transparently.

One promising direction for future work is input test case validation. While the current implementation supports question-answering, the system could be extended to accept sets of test cases as input and assess them against established criteria in combinatorial testing. For example, the Oracle could evaluate whether a given suite of test inputs provides sufficient coverage of parameter interactions or adheres to practices like 2-way or 3-way testing. This functionality

would allow the Oracle to not only explain testing theory, but also to help engineers assess the quality of their test designs in real time—bridging the gap between theoretical guidance and practical validation [23], [24].

The broader significance of this project lies in the integration of domain knowledge into an accessible interface. By adapting combinatorial testing principles to different fields, the Oracle has the potential to support decision-making in areas beyond software engineering. As illustrated in Figure 6-9, the system could be adapted to identify effective combinations in e-commerce product testing, where multiple user-facing parameters interact. This kind of domain transfer is achievable through the same RAFT pipeline, highlighting the system's flexibility and relevance across disciplines.

As language models continue to evolve, embedding expert reasoning patterns and literature-backed validation workflows will be essential for building trustworthy tools. The DoE Oracle demonstrates that such a system can be both specialized and practical—designed not just to answer questions, but to support engineers in applying structured testing methods to real-world challenges.

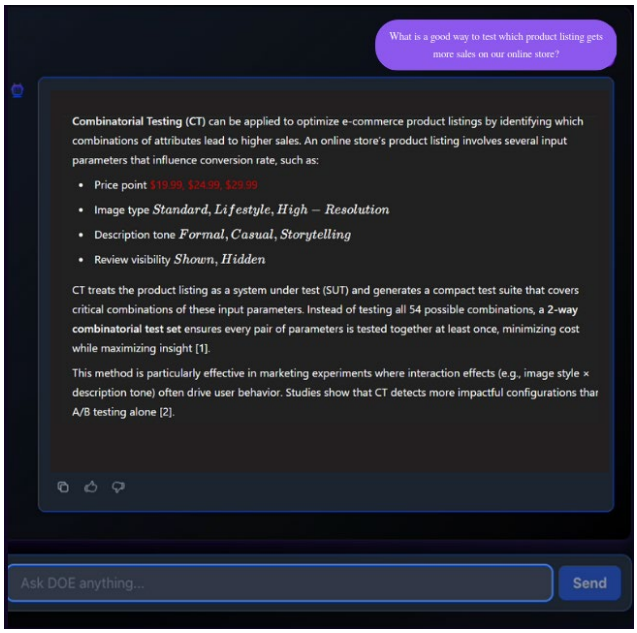

Figure 9. Domain-specific application of combinatorial testing. This example illustrates how combinatorial testing can be adapted to evaluate product listing configurations in e-commerce. Parameters such as price, image style, and description tone are treated as variables in a system under test, and a reduced test suite is applied to uncover impactful combinations while limiting the total number of tests.

REFERENCES

[1] D. C. Montgomery, *Design and analysis of experiments*, 9th ed. John Wiley & Sons, 2017.

[2] M. Tanco, E. Viles, L. Ilzarbe, and M. J. Alvarez, "Barriers to Design of Experiments in small and medium-sized enterprises," *Quality and Reliability Engineering International*, vol. 25, no. 7, pp. 809-821, 2009. DOI: 10.1002/qre.1022.

[3] X. Li, R. Gao, W. E. Wong, C. Yang, and D. Li, "Applying Combinatorial Testing in Industrial Settings," in Proc. *2016 IEEE Int. Conf. Software Quality, Reliability and Security (QRS)*, Vienna, Austria, 2016, pp. 53-60. DOI: 10.1109/QRS.2016.16.

[4] L. Hu, W. E. Wong, D. R. Kuhn, and R. N. Kacker, "How does combinatorial testing perform in the real world: an empirical study," *Empirical Software Engineering*, vol. 25, pp. 2661–2693, 2020. DOI: 10.1007/s10664-019-09799-2.

[5] A. Vaswani et al., "Attention Is All You Need," in *Proc. Advances in Neural Information Processing Systems 30 (NIPS)*, Long Beach, CA, USA, 2017, pp. 5998-6008. [Available: arXiv:1706.03762.]

[6] Yi, G., Chen, Z., Chen, Z., Wong, W.E. and Chau, N., Exploring the capability of ChatGPT in test generation. In *2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion (QRS-C)* (pp. 72-80). IEEE, 2023.

[7] Liu, P., Chen, Z., Li, Y. and Wong, W.E. Evaluating Large Language Models Via Multi-Modal User Knowledge Graphs: A Comprehensive Assessment Framework. In *2025 11th International Symposium on System Security, Safety, and Reliability (ISSSR)* (pp. 278-285). IEEE, 2025

[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proc. Advances in Neural Information Processing Systems 33 (NeurIPS)*, Vancouver, BC, Canada, 2020, pp. 9459-9474. [Online]. Available: arXiv:2005.11401.

[9] D. Li, L. Hu, R. Gao, W. E. Wong, D. R. Kuhn, and R. N. Kacker, "Improving MC/DC and Fault Detection Strength Using Combinatorial Testing," in Proc. *2017 IEEE Int. Conf. Software Quality, Reliability and Security (Companion) (QRS-C)*, Prague, Czech Republic, 2017, pp. 297-303. DOI: 10.1109/QRS-C.2017.131.

[10] Y. Zhang, J. Sun, X. Li, and A. Ritter, "Few-shot learning with GPT models for commonsense reasoning," in Proc. *Findings of the Association for Computational Linguistics*: ACL 2021, Online, 2021, pp.3401–3411.[Online]. Available: https://aclanthology.org/2021.findings-acl.299

[11] V. Braun and V. Clarke, "Using thematic analysis in psychology," Qualitative Research in Psychology, vol. 3, no.2, pp. 77- 101, 2006. DOI: 10.1191/1478088706qp063oa.

[12] M. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, and Y. Choi, "Defending Against Neural Fake News," in Proc. *Advances in Neural Information Processing Systems 32 (NeurIPS)*, Vancouver, BC, Canada, 2019, pp. 9054–9065. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/3e9f0fc9b2f89e043bc6233994dfcf76-Paper.pdf

[13] S. Zhao, D. Balasubramanian, D. Khashabi, and D. Roth, "Negative examples improve information retrieval for open-domain QA," in *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Online, 2021, pp. 764–773.

[14] K. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih, "Dense passage retrieval for open-domain question answering," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Online, 2020, pp. 6769–6781.

[15] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535– 547, 2021. DOI: 10.1109/TBDATA.2019.2921572.

[16] J. Gu et al., "InstructDial: Improving Dialogue Generation with Task-specific Supervision," in Proc. *EMNLP*, Abu Dhabi, United Arab Emirates, 2022, pp. 776- 790. [Online]. Available: https://aclanthology.org/2022.emnlp-main.55

[17] Apple Inc., "MLX: Machine learning on Apple Silicon," *Apple Machine Learning Research*, 2023. [Online]. Available: https://machinelearning.apple.com/research/mlx

[18] Y. Liu, Z. Meng, Y. Zheng, and J. Li, "Contrastive learning for distantly supervised open-domain question answering," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Punta Cana, Dominican Republic, 2021. [Online]. Available: https://arxiv.org/abs/2107.00414

[19] P. C. Jorgensen, *Software Testing: A Craftsman's Approach*, 4th ed., Boca Raton, FL: CRC Press, 2013.

[20] IEEE Std 29119-3-2013, *Software and Systems Engineering—Software Testing—Part 3: Test Documentation*, IEEE, 2013.

[21] D. Ji, X. Wang, Z. Zhang, Y. Liu, J. Sun, and W. Che, "Survey of hallucination in natural language generation*," ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–38, 2023.

[22] M. Borgeaud et al., "Improving language models by retrieving from trillions of tokens," in *Proc. 39th Int. Conf. Machine Learning (ICML)*, Baltimore, MD, USA, Jul. 2022, vol. 162, pp. 2206–2240.

[23] A. Ram, A. Newell, and M. Collins, "Conversational AI: The science behind the tools," *IBM Journal of Research and Development*, vol. 64, no. 2/3, pp. 1–12, Mar./Jun. 2020.

[24] D. R. Kuhn, D. R. Wallace, and A. M. Gallo, "Software fault interactions and implications for software testing," *IEEE Trans. Softw. Eng.*, vol. 30, no. 6, pp. 418–421, Jun. 2004.

[25] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun, and J. Lawrence, "IPOG: A general strategy for t-way software testing," in *Proc. 14th Annu. IEEE Int. Conf. Eng. Comput.-Based Syst.*, Tucson, AZ, USA, 2007, pp. 549–556.