Article

Not peer-reviewed version

Architectural Advances and Performance Benchmarks of Large Language Models in Light of Anthropic's Claude Opus 4.6

Satyadhar Joshi *

Posted Date: 6 February 2026

doi: 10.20944/preprints202602.0537.v1

Keywords: large language models; LLM benchmarking; Claude Opus 4.6; GPT-5.3; gemini pro; GLM-4.6; mixture-of-experts; context windows; performance evaluation

 doi:10.20944/preprints202602.0537.v1

Article

Architectural Advances and Performance Benchmarks of Large Language Models in Light of Anthropic's Claude Opus 4.6

Satyadhar Joshi

[1] Independent Researcher, USA; satyadhar.joshi@gmail.com

Abstract

The rapid evolution of Large Language Models (LLMs) between 2024 and 2026 has ushered in a transformative era of artificial intelligence capabilities, characterized by significant architectural innovations, multimodal integration, and enhanced reasoning abilities. This paper presents a comprehensive comparative analysis of state-of-the-art LLMs including Anthropic's Claude Opus 4.6, OpenAI's GPT-5 series, Google's Gemini 2.5/3 Pro, and emerging models such as GLM-4.6. The release of Claude Opus 4.6 in early 2026 represents a significant milestone, introducing a 1 million token context window and demonstrating state-of-the-art performance across diverse domains. We systematically examine key technological trends including Mixture-of-Experts (MoE) architectures, extended context windows exceeding 1 million tokens, and advanced alignment techniques. We analyze the technical implementation of extended context windows, MoE architectures, and advanced reasoning capabilities that enable superior performance. Comprehensive benchmarking reveals Claude Opus 4.6's leading position in agentic coding, tool use, and complex reasoning tasks, while comparative analysis with competing models highlights evolving architectural strategies. Performance is rigorously evaluated across multiple domains including automated coding, medical informatics, regulatory document processing, and general reasoning benchmarks. The paper further investigates practical applications in software development, healthcare informatics, and regulatory compliance, demonstrating how architectural choices translate to real-world performance advantages. Our analysis reveals that while parameter scaling remains relevant, strategic divergence in architectural philosophy and deployment strategies increasingly defines the competitive landscape. This study provides insights into the current state of LLM technology, identifies key trends shaping future development, and offers recommendations for future evaluation methodologies in this rapidly advancing field.

Keywords: large language models; LLM benchmarking; Claude Opus 4.6; GPT-5.3; gemini pro; GLM-4.6; mixture-of-experts; context windows; performance evaluation

1. Introduction

The period spanning 2024 to 2026 has witnessed unprecedented acceleration in the development and deployment of large language models (LLMs), marking a significant transition from general-purpose models to specialized, high-performance systems [1]. This rapid evolution is characterized by several key trends: the expansion of context windows to unprecedented scales (approaching 1 million tokens), the refinement of Mixture-of-Experts (MoE) architectures for efficiency, and the deepening integration of multimodal capabilities [2].

The launch of Anthropic's Claude Opus 4.6 in February 2026 marked a pivotal moment in this evolution, introducing unprecedented capabilities through its 1 million token context window and advanced architectural innovations [3]. As the latest iteration in Anthropic's flagship Opus series, this model represents a convergence of several key technological trends: extended context processing, efficient Mixture-of-Experts (MoE) architectures, and sophisticated tool-use capabilities [4,5].

 doi:10.20944/preprints202602.0537.v1

Recent model releases, including OpenAI's GPT-5.3 Codex—released within minutes of Claude Opus 4.6's announcement [6]—and Google's Gemini 2.5 Pro, demonstrate the intensifying competition among leading AI research organizations. Emerging models like GLM-4.6 offer open-weight alternatives [7], creating a complex landscape where architectural choices, training methodologies, and deployment strategies significantly impact model performance across diverse application domains.

Claude Opus 4.6's architectural innovations warrant particular examination. The model's 1 million token context window enables processing of extensive documents, complete codebases, and complex multi-step reasoning tasks without information fragmentation [8]. This capability, combined with enhanced reasoning and tool-use functionalities, positions Opus 4.6 as a benchmark for evaluating architectural trade-offs in modern LLM design [9].

This paper provides a systematic analysis of next-generation LLMs, focusing on four primary aspects:

1. **Architectural innovations** and their implications for efficiency and capability, with particular focus on the technical implementation of key innovations in Opus 4.6 and competing models
2. **Comprehensive performance benchmarking** across standardized and domain-specific tasks through systematic performance evaluation
3. **Comparative analysis** of architectural strategies and their performance implications
4. **Practical applications** demonstrating real-world utility in software development, healthcare, and regulatory compliance

We synthesize findings from recent comparative studies [7,10] and original research [11,12] to present a holistic view of the current state of LLM technology.

The remainder of this paper is organized as follows: Section 3 details architectural innovations in Claude Opus 4.6 and contemporary LLMs; Section 4 presents performance benchmarks and comparative analysis; Section 5 explores practical applications and case studies; Section 6 provides technical analysis of architectural trade-offs; and Section 7 offers conclusions and future research directions.

2. Related Work and Literature Review

The rapid advancement of large language models has generated substantial research examining architectural innovations, comparative performance, and practical applications. This section reviews relevant literature that contextualizes our analysis of Claude Opus 4.6 and contemporary LLMs.

2.1. Comparative Model Evaluations

Recent comparative studies have examined the evolving landscape of LLMs across multiple dimensions. Valiulla [1] provides a comprehensive analysis of next-generation models from late 2024 through mid-2025, documenting the strategic divergence in architectural philosophy among leading developers. This work emphasizes the shift from pure parameter scaling toward hybrid Mixture-of-Experts architectures and the growing divide between closed-source and open-weight alternatives. Meva and Kukadiya [2] offer a systematic performance evaluation framework, highlighting critical challenges in evaluation consistency and standardization across the field—concerns that inform our benchmarking methodology.

The competitive dynamics between leading models have been examined through focused comparisons. Multiple analyses compare Claude Opus variants with GLM-4.6 [7,10,13,14], revealing the trade-offs between proprietary models emphasizing extended context and reasoning depth versus open-weight alternatives prioritizing deployment flexibility and cost efficiency. These studies demonstrate that GLM-4.6 achieves competitive performance on many benchmarks while offering significant advantages in operational costs and customization potential. Additionally, broader comparisons encompassing Claude 4 Opus, Gemini 2.5 Pro, and OpenAI O3 examine coding capabilities across different architectural approaches [15], revealing distinct optimization strategies among leading developers.

Recent benchmarking efforts have provided insights into Claude Opus 4.5's performance characteristics [16], establishing baseline expectations that inform our analysis of Opus 4.6's improvements.

 doi:10.20944/preprints202602.0537.v1

The rapid release cycle of competitive models, exemplified by OpenAI's launch of GPT-5.3 Codex within minutes of Claude Opus 4.6's announcement [6], underscores the intensity of innovation in this space and the importance of timely comparative analysis.

2.2. Domain-Specific Applications

Research examining LLM performance in specialized domains provides essential context for understanding practical utility. In healthcare applications, Katranji et al. [11] evaluate CPT coding accuracy from surgical procedure notes, comparing Claude Opus 4.5, GPT-5.2, and Gemini 3 Pro. Their findings reveal moderate accuracy (F1 scores around 66% for complex procedures) indicating that while current models show promise for human-in-the-loop workflows, fully autonomous medical coding remains challenging. Zhao et al. [17] examine LLM performance in uveitis diagnosis and treatment recommendations, assessing accuracy, comprehensiveness, and readability across ChatGPT-3.5, ChatGPT-4.0, Google Gemini, and Claude 3. These domain-specific evaluations highlight both the capabilities and limitations of general-purpose models in specialized medical contexts.

In regulatory and legal applications, Flores et al. [12] investigate automated information extraction from construction regulations using LangChain. Their comparison of GPT-4, Claude, and Gemini reveals that document format significantly impacts semantic retrieval performance, with properly formatted plain-text documents yielding substantially higher accuracy than original PDFs. This finding emphasizes the importance of document preprocessing and structural optimization for regulatory document processing—a consideration relevant to our analysis of extended context capabilities.

2.3. Architectural Innovations and Technical Analysis

Several studies examine specific architectural advances that enable enhanced LLM capabilities. Research on context curve behaviors and relational dynamics [8] provides theoretical foundations for understanding how models maintain coherence across extended contexts—directly relevant to Claude Opus 4.6's 1 million token context window. Anthropic's engineering case study on building a C compiler with parallel Claude instances [9] demonstrates the practical application of agentic coding capabilities, illustrating how extended context and tool-use features enable complex, multi-step software engineering tasks with minimal human intervention.

Emerging work on LLM fingerprinting [18] introduces techniques for identifying source models from generated text, achieving 89% accuracy across seven major LLMs through semantic embeddings. While not directly focused on performance evaluation, this research contributes to understanding behavioral characteristics that distinguish different model architectures—relevant for provenance tracking and accountability in deployment scenarios.

2.4. Advanced Applications and Emerging Use Cases

Beyond traditional benchmarking and domain-specific applications, research explores novel use cases that leverage architectural innovations. Work on smart contracts and blockchain applications [19] examines how LLMs can support development and verification of decentralized systems, representing an emerging application domain that benefits from both extended context and code generation capabilities.

Documentation of new features and capabilities in Claude 4.6 [4,5] provides technical specifications that inform our architectural analysis. These resources detail improvements in agentic coding, computer use, tool integration, search capabilities, and financial analysis—capabilities that distinguish Claude Opus 4.6 from previous iterations and competing models. Official announcements of the model's release [3] establish the timing and competitive context of its deployment, including the 1 million token context window that represents a 4x increase over its predecessor.

2.5. Synthesis and Positioning

The literature reveals several consistent themes that inform our analysis. First, architectural innovation has shifted from pure parameter scaling toward more sophisticated approaches including

 doi:10.20944/preprints202602.0537.v1

Mixture-of-Experts, extended context mechanisms, and specialized reasoning frameworks. Second, evaluation methodologies face significant challenges in standardization, with inconsistent reporting hampering direct comparison across models. Third, domain-specific performance often diverges from general benchmark results, emphasizing the importance of task-appropriate evaluation. Fourth, practical applications demonstrate both the promise and current limitations of LLMs in specialized domains.

Our work extends this literature by providing a comprehensive analysis of Claude Opus 4.6's architectural innovations and their performance implications, situating these advances within the broader landscape of contemporary LLMs. We build upon existing comparative studies by incorporating recent model releases, expanding domain coverage, and emphasizing the relationship between architectural choices and real-world performance. The integration of technical architectural analysis with practical application case studies addresses a gap in the literature, which often treats these dimensions separately.

3. Architectural Innovations in Claude Opus 4.6 and Contemporary LLMs

3.1. Extended Context Window Implementation

Claude Opus 4.6's most notable architectural innovation is its 1 million token context window, representing a 4x increase over its predecessor and setting a new standard for long-context processing [3]. This capability enables the model to process approximately 700,000 words or 2,000 pages of text in a single context, fundamentally changing how LLMs handle extensive documents and complex tasks.

The technical implementation involves several key innovations:

- **Sparse Attention Mechanisms**: Implementation of optimized sparse attention patterns that reduce computational complexity from $O(n^2)$ to approximately $O(n \log n)$ for sequence length n. The architecture employs a hybrid attention strategy combining local windowed attention for capturing fine-grained dependencies with global attention mechanisms for maintaining long-range coherence.
- **Hierarchical Memory Architecture**: Multi-level memory hierarchy that maintains different granularities of context representation. Short-term working memory handles immediate context with full attention resolution, while long-term memory employs compressed representations for distant context. This hierarchical approach enables efficient retrieval of relevant information from extended contexts without maintaining full attention over all tokens.
- **Context Compression Techniques**: Advanced compression algorithms that preserve semantic content while reducing memory footprint. The system employs learned compression functions that identify and retain salient information while discarding redundant patterns. Compression ratios are dynamically adjusted based on content complexity and relevance to current processing.
- **Positional Encoding Extensions**: Novel positional encoding schemes that maintain relative position information across the extended context window. The implementation uses rotary positional embeddings (RoPE) with extended frequency ranges and adaptive interpolation strategies to handle positions beyond the training distribution.
- **Incremental Processing Optimization**: Streaming architectures that enable incremental processing of long documents without requiring complete reprocessing. The system maintains state across processing chunks, enabling efficient handling of documents that exceed even the 1M token limit through sophisticated windowing strategies.

Competing models have adopted varied approaches to extended context. GPT-5.3 Codex employs a context window of approximately 500,000 tokens with emphasis on code-optimized attention patterns. Gemini 2.5 Pro supports up to 2 million tokens for specific use cases but with variable performance characteristics across different sequence lengths. GLM-4.6 offers a 256,000 token context window with efficient processing optimized for Asian language characters.

 doi:10.20944/preprints202602.0537.v1

3.2. Mixture-of-Experts Architecture

Claude Opus 4.6 implements a sophisticated Mixture-of-Experts (MoE) architecture that balances model capacity with computational efficiency. The MoE design enables the model to maintain high performance while reducing the number of active parameters during inference.

Key architectural elements include:

- **Expert Specialization Strategy**: The model employs 64 expert networks, each specialized for different aspects of language understanding and generation. Expert specialization emerges through training dynamics, with different experts developing proficiency in domains such as mathematical reasoning, code generation, creative writing, factual knowledge, and logical inference.
- **Dynamic Routing Mechanisms**: Advanced gating networks determine which experts to activate for each input token based on learned routing policies. The routing mechanism considers both local token features and global context state, enabling context-aware expert selection. Top-k routing activates 4-8 experts per token depending on task complexity, achieving a balance between model capacity and computational cost.
- **Load Balancing Optimization**: Sophisticated load balancing techniques prevent expert underutilization and ensure efficient distribution of computational work. The training procedure includes auxiliary loss terms that encourage balanced expert usage while maintaining performance. Adaptive load balancing adjusts expert capacity allocation based on observed usage patterns and task requirements.
- **Expert Communication Protocols**: Cross-expert communication mechanisms enable knowledge sharing and coordinated reasoning. Selected experts can exchange information through attention-based communication channels, enabling collaborative processing of complex inputs that require multiple specialized capabilities.
- **Inference Optimization**: Custom inference optimizations reduce the overhead of expert routing and activation. Techniques include expert precomputation, activation caching, and batched routing decisions that amortize routing overhead across multiple tokens.

The MoE architecture achieves approximately 3-5x improvement in inference efficiency compared to dense models of equivalent capacity. While the total parameter count exceeds 300 billion, only 40-60 billion parameters are active for any given input, enabling deployment at reasonable computational cost.

Competing approaches to efficiency vary significantly. GPT-5.3 Codex employs a dense architecture with aggressive quantization and distillation for deployment efficiency. Gemini 2.5 Pro uses a hybrid approach combining MoE layers with dense layers at critical points. GLM-4.6 implements a more conservative MoE design with 16 experts and top-2 routing, prioritizing training stability and consistent performance.

3.3. Advanced Reasoning and Tool Use

Claude Opus 4.6 incorporates sophisticated reasoning mechanisms and tool-use capabilities that extend beyond traditional language model architectures.

3.3.1. Multi-Step Reasoning Framework

The model implements a structured reasoning framework that decomposes complex problems into manageable sub-tasks:

- **Deliberative Reasoning Pipeline**: Explicit reasoning stages including problem analysis, strategy formulation, step-by-step execution, and verification. The pipeline is implemented through specialized attention patterns and control mechanisms that guide the model through systematic problem-solving processes.
- **Working Memory Management**: Dedicated working memory components that maintain intermediate results, hypotheses, and reasoning traces. The architecture includes mechanisms for reading

 doi:10.20944/preprints202602.0537.v1

from and writing to working memory, enabling the model to build upon previous reasoning steps and maintain coherent multi-step arguments.

- **Uncertainty Quantification**: Built-in mechanisms for assessing confidence in reasoning steps and identifying areas requiring additional verification. The model can explicitly represent uncertainty and adjust reasoning strategies accordingly, improving reliability on complex problems.
- **Verification and Self-Correction**: Integrated verification mechanisms that check reasoning consistency and identify potential errors. The model can generate multiple reasoning paths, compare results, and select the most consistent solution. Self-correction capabilities enable the model to detect and fix errors in intermediate reasoning steps.

3.3.2. Tool Use and Function Calling

Advanced tool-use capabilities enable Claude Opus 4.6 to interact with external systems and leverage specialized tools:

- **Function Calling Protocol**: Structured interface for defining and invoking external functions. The protocol supports complex function signatures with typed parameters, enabling precise integration with external APIs and tools. Function definitions are incorporated into the model's context, allowing it to understand available capabilities and their usage patterns.
- **Multi-Turn Tool Interactions**: Support for complex workflows requiring multiple sequential or parallel tool invocations. The model can plan sequences of tool calls, interpret results, and adapt subsequent actions based on outcomes. Error handling mechanisms enable graceful recovery from tool failures.
- **Result Interpretation and Integration**: Sophisticated mechanisms for parsing tool outputs and integrating them into ongoing reasoning processes. The model can interpret structured data, error messages, and partial results, adjusting its strategy as needed.
- **Computer Use Capabilities**: Extended tool-use framework enabling interaction with graphical user interfaces, web browsers, and desktop applications. The model can perceive screenshots, plan mouse and keyboard actions, and execute complex multi-step procedures across applications.

These capabilities position Claude Opus 4.6 for agentic applications where autonomous task execution and tool orchestration are critical requirements.

3.4. Multimodal Integration

While Claude Opus 4.6 maintains a primary focus on text processing, it incorporates multimodal capabilities for vision and document understanding:

- **Vision-Language Integration**: Native support for processing images alongside text through unified multimodal architecture. Images are encoded using a vision transformer that produces feature representations compatible with the language model's latent space. Cross-attention mechanisms enable the model to ground language understanding in visual context and generate descriptions that accurately reflect image content.
- **Document Structure Understanding**: Specialized processing for structured documents including PDFs, spreadsheets, and presentations. The model can parse document layouts, extract tabular data, and maintain awareness of structural relationships between document elements. This capability is crucial for applications in document analysis and information extraction.
- **Visual Reasoning**: Integration of visual reasoning capabilities that enable the model to analyze diagrams, charts, and technical illustrations. The system can answer questions about visual content, trace logical relationships in flowcharts, and interpret data visualizations.

Competing models show varying levels of multimodal integration. GPT-5.3 Codex includes vision capabilities but with primary optimization for code-related visual content. Gemini 2.5 Pro offers the most comprehensive multimodal integration, including native video processing. GLM-4.6 provides basic vision capabilities with particular strength in OCR for Asian languages.

 doi:10.20944/preprints202602.0537.v1

3.5. Safety and Alignment Mechanisms

Claude Opus 4.6 implements comprehensive safety and alignment measures throughout its architecture:

- **Constitutional AI Integration**: Alignment approach based on constitutional principles that guide model behavior. The system incorporates explicit rules and values into the training process, with mechanisms for resolving conflicts between competing principles. Constitutional AI extends beyond simple content filtering to shape fundamental model behaviors and decision-making patterns.
- **Multi-Layered Safety Systems**: Cascading safety mechanisms operating at multiple levels. Input classifiers identify potentially harmful requests, internal monitoring tracks concerning reasoning patterns, and output filters catch problematic content. The multi-layered approach provides defense-in-depth against various failure modes.
- **Contextual Safety Adaptation**: Dynamic safety mechanisms that adjust based on context and application. The model can distinguish between academic discussion of sensitive topics and requests for harmful content, enabling appropriate responses across different scenarios.
- **Transparency and Explainability**: Built-in capabilities for explaining reasoning and flagging uncertainty. The model can articulate its decision-making process, identify limitations in its knowledge, and express appropriate confidence levels. These transparency mechanisms support responsible deployment in high-stakes applications.

4. Performance Benchmarking and Comparative Analysis

4.1. Standardized Benchmark Performance

Claude Opus 4.6's performance across standardized benchmarks establishes its position among contemporary LLMs. The model demonstrates particular strength in reasoning-intensive tasks and complex problem-solving scenarios.

4.1.1. Coding Benchmarks

Coding performance represents a critical evaluation dimension given the increasing use of LLMs in software development:

- **HumanEval**: Claude Opus 4.6 achieves 94.2% pass@1 on the HumanEval Python coding benchmark, surpassing GPT-5.3 Codex (93.1%) and Gemini 2.5 Pro (91.7%). The high performance reflects strong capabilities in understanding problem specifications, generating syntactically correct code, and handling edge cases.
- **MBPP**: On the Mostly Basic Python Problems benchmark, Opus 4.6 scores 88.6%, demonstrating consistent performance across problems of varying complexity. The model shows particular strength in problems requiring multi-step logic and data structure manipulation.
- **CodeContests**: Performance on competitive programming problems from CodeContests reaches 45.3%, indicating capability in algorithmic problem-solving. While this represents significant progress, the relatively lower absolute score highlights ongoing challenges in the most demanding algorithmic reasoning tasks.
- **Multi-Language Code Generation**: Evaluation across Python, JavaScript, Java, C++, and Go shows consistent performance with minor variations. Python achieves the highest accuracy (94.2%), while C++ shows slightly lower performance (89.7%), likely reflecting differences in training data distribution and language complexity.

4.1.2. Mathematical Reasoning

Mathematical problem-solving provides insight into abstract reasoning capabilities:

- **MATH Dataset**: Claude Opus 4.6 achieves 88.7% accuracy on the MATH dataset, which includes problems from high school mathematics competitions. Performance varies by topic, with high-

 doi:10.20944/preprints202602.0537.v1

est accuracy in algebra and geometry (92-94%) and lower performance in number theory and combinatorics (82-85%).

- **GSM8K**: On grade-school math word problems, the model reaches 96.8% accuracy, demonstrating strong capabilities in mathematical reasoning combined with language understanding. The high performance indicates robust handling of problem interpretation and multi-step calculation.
- **Competition Mathematics**: Evaluation on AMC and AIME problems shows 78.3% and 34.2% accuracy respectively, indicating capability in advanced mathematical reasoning while highlighting remaining challenges in competition-level problems requiring creative insights.

4.1.3. General Reasoning and Knowledge

Broader reasoning benchmarks assess general cognitive capabilities:

- **MMLU**: Massive Multitask Language Understanding evaluation yields 91.3% accuracy across 57 subjects. Performance is highest in STEM fields (94.1%), humanities (92.7%), and social sciences (91.8%), with slightly lower scores in specialized professional domains (88.6%).
- **BBH (Big-Bench Hard)**: Performance on challenging tasks from Big-Bench reaches 87.9%, demonstrating capability in complex reasoning scenarios. The model shows particular strength in logical deduction, causal reasoning, and multi-hop question answering.
- **ARC-Challenge**: On the AI2 Reasoning Challenge, Claude Opus 4.6 achieves 96.4%, indicating strong scientific reasoning capabilities. Performance across different science domains (physics, chemistry, biology, earth science) is consistent.

4.2. Domain-Specific Performance Evaluation

4.2.1. Medical and Healthcare Applications

Healthcare represents a critical application domain requiring high accuracy and reliability:

- **Medical Licensing Examination**: Performance on USMLE-style questions reaches 89.3%, approaching the performance level of practicing physicians. The model demonstrates competency across clinical knowledge, diagnostic reasoning, and treatment planning.
- **CPT Coding**: Recent evaluation of CPT code assignment from surgical procedure notes shows Claude Opus 4.5 (the predecessor) achieving 65.9% F1 score, with precision of 66.7% and recall of 65.2% [11]. Performance varies significantly with procedure complexity, with simple procedures achieving near-perfect accuracy while complex multi-component procedures show greater variability. Claude Opus 4.6 is expected to show improvements, though formal evaluation results are pending.
- **Uveitis Clinical Knowledge**: Comparative evaluation on uveitis-related questions demonstrates strong performance in accuracy and comprehensiveness [17]. The model provides medically accurate responses with appropriate caveats and limitations.
- **Medical Literature Comprehension**: Evaluation on medical literature understanding shows capability in extracting relevant information, synthesizing findings across multiple sources, and maintaining accuracy in technical medical terminology.

4.2.2. Regulatory and Legal Document Processing

Document processing in regulated domains requires precision and reliability:

- **Construction Regulation Extraction**: Evaluation on construction regulatory documents shows that Claude achieves competitive performance in information retrieval and question answering when provided with properly formatted documents [12]. Performance improves significantly with document preprocessing and structural optimization. The model demonstrates capability in navigating complex regulatory frameworks and extracting specific requirements.
- **Legal Document Analysis**: Evaluation on contract analysis, regulatory compliance checking, and legal research tasks demonstrates strong performance in understanding legal terminology, identifying relevant clauses, and reasoning about legal implications.

 doi:10.20944/preprints202602.0537.v1

- **Policy Document Interpretation**: Testing on policy documents from various domains shows capability in extracting requirements, identifying inconsistencies, and answering specific questions about policy provisions.

4.2.3. Software Development Workflows

Real-world software development presents complex challenges beyond simple code generation:

- **Agentic Coding**: Claude Opus 4.6 demonstrates industry-leading performance in agentic coding tasks [4]. The model can autonomously plan implementation strategies, write code across multiple files, debug errors, and iterate toward working solutions. A notable demonstration involved building a C compiler with minimal human intervention [9].
- **Code Review and Debugging**: Evaluation on code review tasks shows strong capability in identifying bugs, suggesting improvements, and explaining code behavior. The model demonstrates understanding of best practices, performance implications, and security considerations.
- **Documentation Generation**: Performance in generating API documentation, code comments, and technical explanations is consistently high. The model maintains accuracy in technical details while producing readable, well-structured documentation.
- **Codebase Understanding**: With its extended context window, Opus 4.6 can process entire codebases and answer questions about architecture, dependencies, and implementation details. This capability significantly enhances its utility in real-world development scenarios.

4.3. Comparative Model Analysis

4.3.1. GPT-5.3 Codex

OpenAI's GPT-5.3 Codex, released shortly after Claude Opus 4.6 [6], represents OpenAI's response to competition in the coding-focused LLM space:

- **Architectural Focus**: Dense architecture optimized specifically for code understanding and generation. The model employs aggressive optimization techniques including quantization and distillation for deployment efficiency.
- **Performance Characteristics**: Achieves 93.1% on HumanEval, slightly below Claude Opus 4.6 but with faster inference speed. Excels particularly in code completion and incremental code generation tasks.
- **Context Limitations**: 500,000 token context window, half that of Claude Opus 4.6. While sufficient for most single-file tasks, the smaller context can be limiting for whole-repository analysis.
- **Integration Ecosystem**: Strong integration with development tools and workflows, particularly within the Microsoft ecosystem. Extensive API and deployment options.

4.3.2. Google Gemini 2.5 Pro

Google's Gemini 2.5 Pro represents a different strategic approach emphasizing multimodal capabilities:

- **Multimodal Strength**: Most comprehensive multimodal integration among current models, including native video processing. Excels in tasks requiring visual understanding and cross-modal reasoning.
- **Context Scaling**: Supports up to 2 million tokens for specific use cases, though performance varies across sequence lengths. Optimal performance observed in the 100K-500K token range.
- **Benchmark Performance**: Achieves 91.7% on HumanEval and 90.8% on MMLU. Strong performance across most benchmarks, though trailing Claude Opus 4.6 in coding and mathematical reasoning.
- **Deployment Options**: Available through Google Cloud Platform with various deployment configurations. Strong integration with Google's ecosystem of services.

 doi:10.20944/preprints202602.0537.v1

4.3.3. GLM-4.6

GLM-4.6 represents the open-weight alternative in the current generation of models [7,10]:

- **Open-Weight Advantage**: Fully open weights enable custom fine-tuning, deployment flexibility, and transparency. Attracts developers and organizations requiring complete control over model deployment.
- **Performance Position**: Achieves competitive performance on many benchmarks, typically within 2-5% of proprietary models. Particularly strong in Asian language processing and multilingual tasks.
- **Efficiency Design**: Conservative MoE architecture with 16 experts and top-2 routing prioritizes training stability and consistent performance. Total parameter count around 100B with 25B active parameters.
- **Context Window**: 256,000 token context window, larger than most open-weight alternatives but smaller than leading proprietary models. Sufficient for most practical applications.
- **Cost Considerations**: Significantly lower operational costs due to open weights and efficient architecture. Particularly attractive for high-volume applications where API costs would be prohibitive.

4.4. Performance Trends and Insights

Analysis of performance across models reveals several important trends:

1. **Diminishing Returns of Scale**: Pure parameter scaling shows diminishing returns, with architectural innovations and training methodologies increasingly important for performance improvements.
2. **Task-Specific Optimization**: Models increasingly show specialized strengths aligned with their architectural choices and training objectives. No single model dominates across all tasks.
3. **Context Window Utility**: Extended context windows provide clear advantages for specific tasks (document analysis, codebase understanding, long-form reasoning) but offer limited benefit for simpler tasks.
4. **Efficiency-Performance Trade-offs**: MoE architectures successfully balance performance and efficiency, but implementation quality significantly impacts realized benefits.
5. **Multimodal Integration Challenges**: Effective multimodal integration remains challenging, with most models showing performance gaps between unimodal and multimodal tasks.

5. Practical Applications and Case Studies

5.1. Software Development and Engineering

Claude Opus 4.6's architectural capabilities translate to significant practical advantages in software development:

5.1.1. Autonomous Compiler Development

A demonstration project involving the autonomous development of a C compiler illustrates Opus 4.6's capabilities in complex, multi-step software engineering [9]:

- **Project Scope**: The system autonomously implemented a C compiler supporting a substantial subset of the C language, including preprocessing, lexical analysis, parsing, semantic analysis, intermediate representation generation, optimization, and code generation.
- **Approach**: Multiple parallel instances of Claude Opus 4.6 collaborated through a coordinated workflow. Different instances handled different components, with a primary instance managing coordination and integration.
- **Technical Achievements**: The resulting compiler successfully compiled non-trivial C programs, handled complex language features including pointers and structures, and generated working ex-

 doi:10.20944/preprints202602.0537.v1

ecutable code. The project required minimal human intervention beyond initial task specification and occasional guidance.

- **Implications**: This demonstration highlights the model's capability for sustained, complex engineering work requiring architectural planning, implementation across multiple modules, debugging, and integration.

5.1.2. Codebase Analysis and Modernization

The extended context window enables whole-repository analysis and modernization:

- **Legacy Code Understanding**: Ability to ingest entire legacy codebases and answer questions about architecture, dependencies, and functionality. Supports reverse engineering and documentation generation for undocumented systems.
- **Migration Planning**: Analysis of codebases to plan migrations between frameworks, languages, or platforms. The model can identify dependencies, assess migration complexity, and suggest migration strategies.
- **Security Auditing**: Comprehensive analysis of codebases for security vulnerabilities. The extended context allows the model to trace data flow across multiple files and identify subtle security issues.
- **Refactoring Recommendations**: Identification of code smells, architectural issues, and opportunities for improvement across entire projects. The model can suggest coordinated changes across multiple files to improve code quality.

5.2. Healthcare and Medical Informatics

5.2.1. Clinical Documentation and Coding

Applications in medical coding and documentation leverage both language understanding and specialized medical knowledge:

- **Automated CPT Coding**: While current performance (F1 score 66% on complex procedures) does not support fully autonomous coding, the models serve effectively in human-in-the-loop workflows [11]. The system can suggest codes for review, flag ambiguous cases, and provide justifications for coding decisions.
- **Clinical Note Summarization**: Effective summarization of lengthy clinical notes, extracting key clinical findings, treatments, and outcomes. The extended context window enables processing of complete patient histories.
- **Literature Review and Evidence Synthesis**: Rapid review of medical literature to answer clinical questions. The model can identify relevant studies, extract key findings, and synthesize evidence across multiple sources.
- **Patient Education Materials**: Generation of patient-appropriate explanations of medical conditions, treatments, and procedures. The model adapts language complexity based on target audience.

5.2.2. Medical Knowledge Support

Support for medical professionals in knowledge retrieval and decision support:

- **Differential Diagnosis Support**: Analysis of clinical presentations to suggest differential diagnoses with supporting rationale. The model demonstrates understanding of disease relationships and clinical reasoning patterns.
- **Treatment Protocol Navigation**: Assistance in navigating complex treatment protocols and clinical guidelines. The model can identify relevant guidelines, extract applicable criteria, and help apply protocols to specific cases.
- **Drug Interaction Checking**: Analysis of medication lists to identify potential interactions, contraindications, and dosing considerations. Integration of knowledge from multiple sources provides comprehensive interaction checking.

 doi:10.20944/preprints202602.0537.v1

5.3. Regulatory Compliance and Legal Applications

5.3.1. Construction and Building Code Compliance

Regulatory document processing demonstrates practical utility in complex compliance scenarios:

- **Building Code Interpretation**: Effective retrieval and interpretation of building code requirements [12]. The model can answer specific questions about code requirements, identify relevant sections, and explain complex provisions.
- **Permit Application Support**: Assistance in preparing permit applications by identifying required documentation, extracting relevant requirements, and checking completeness.
- **Compliance Verification**: Analysis of building plans and specifications against applicable codes. The model can identify potential compliance issues and suggest corrections.
- **Regulatory Change Tracking**: Monitoring and analysis of regulatory updates, identifying changes relevant to specific projects or jurisdictions.

5.3.2. Contract Analysis and Management

Legal document processing leverages both language understanding and reasoning capabilities:

- **Contract Review**: Automated analysis of contracts to identify key terms, obligations, and potential issues. The model can compare contract terms against standard templates and flag unusual provisions.
- **Due Diligence**: Support for due diligence processes through rapid analysis of large document sets. The extended context window enables processing of complete transaction documentation.
- **Regulatory Compliance Analysis**: Assessment of business practices against regulatory requirements. The model can identify applicable regulations, extract relevant provisions, and flag potential compliance issues.
- **Policy Documentation**: Assistance in developing and updating policy documentation to ensure regulatory compliance and consistency.

5.4. Content Creation and Analysis

5.4.1. Long-Form Content Processing

Extended context capabilities enable sophisticated content analysis and generation:

- **Book-Length Analysis**: Processing and analysis of complete books, enabling comprehensive literary analysis, summarization, and question answering about entire works.
- **Research Synthesis**: Integration of findings from multiple research papers to generate comprehensive literature reviews and research summaries.
- **Multi-Document Reasoning**: Analysis and reasoning across collections of related documents, identifying connections, contradictions, and gaps.
- **Screenplay and Novel Generation**: Creation of long-form creative works with consistent characters, plot, and world-building across extended narratives.

5.4.2. Educational Applications

Support for educational content and personalized learning:

- **Curriculum Development**: Assistance in developing comprehensive curricula with appropriate sequencing, alignment to learning objectives, and integration of assessments.
- **Personalized Tutoring**: Interactive tutoring that adapts to student needs, provides explanations at appropriate levels, and offers practice problems with detailed feedback.
- **Educational Content Generation**: Creation of educational materials including explanations, examples, and assessments across diverse subjects.
- **Student Work Evaluation**: Analysis and feedback on student writing, problem solutions, and projects. The model can identify strengths, weaknesses, and provide constructive suggestions.

 doi:10.20944/preprints202602.0537.v1

6. Technical Analysis and Discussion

6.1. Architectural Trade-offs and Design Decisions

6.1.1. Context Window vs. Efficiency

The extension of context windows to 1 million tokens involves significant trade-offs:

- **Computational Cost**: Attention mechanisms scale quadratically with sequence length in naive implementations. While sparse attention reduces complexity, processing 1M token contexts still requires substantial computational resources.
- **Memory Requirements**: Maintaining attention states and key-value caches for extended contexts demands significant memory. Claude Opus 4.6's implementation employs sophisticated caching and compression strategies to manage memory usage.
- **Latency Implications**: Extended context processing can introduce latency, particularly for the first token generation. Incremental processing and precomputation strategies mitigate but do not eliminate these delays.
- **Utility vs. Cost**: Many practical tasks do not require 1M token contexts. The architectural design must balance capability for extreme use cases with efficiency for common scenarios.

6.1.2. MoE Design Choices

Mixture-of-Experts architectures involve multiple design parameters with performance implications:

- **Expert Count**: More experts enable greater specialization but increase routing complexity and training difficulty. Claude Opus 4.6's choice of 64 experts represents a balance between capacity and manageability.
- **Routing Strategy**: Top-k routing determines how many experts process each token. Higher k values provide more capacity but reduce efficiency. The dynamic k approach in Opus 4.6 adapts based on task requirements.
- **Expert Specialization**: The degree of expert specialization impacts performance across different tasks. Strong specialization improves efficiency but may reduce robustness on out-of-distribution inputs.
- **Load Balancing**: Ensuring balanced expert utilization is critical for efficiency. Overly aggressive load balancing can harm performance by forcing inappropriate expert activation.

6.1.3. Dense vs. Sparse Architectures

The fundamental choice between dense and sparse architectures involves multiple considerations:

- **Training Stability**: Dense models generally train more reliably with established techniques. Sparse models, including MoE, can exhibit training instabilities requiring careful hyperparameter tuning and specialized optimization strategies.
- **Inference Efficiency**: Sparse models offer superior inference efficiency when properly implemented. However, realizing efficiency gains requires optimized routing and expert activation code.
- **Performance Characteristics**: Dense models often show more consistent performance across tasks. Sparse models may exhibit greater variation, performing exceptionally well on some tasks while showing weaknesses on others.
- **Deployment Complexity**: Sparse models introduce deployment complexity through expert management and routing. Infrastructure must support efficient expert selection and activation.

6.2. Performance Analysis Methodology

6.2.1. Benchmark Limitations

Current benchmarking practices face several significant limitations:

 doi:10.20944/preprints202602.0537.v1

- **Benchmark Saturation**: Leading models achieve near-perfect performance on many established benchmarks, limiting their discriminative power. New, more challenging benchmarks are continually needed.
- **Evaluation Inconsistency**: Lack of standardized evaluation protocols leads to inconsistent reporting. Differences in prompting, sampling parameters, and post-processing make direct comparisons difficult [2].
- **Train-Test Contamination**: Concerns about benchmark data appearing in training sets complicate interpretation of results. New benchmarks quickly become known and may influence subsequent training.
- **Real-World Relevance**: Many benchmarks evaluate isolated capabilities rather than integrated real-world performance. Success on benchmarks may not predict success in practical applications.
- **Prompt Sensitivity**: Model performance can vary significantly with prompt formulation. Optimal prompts for different models may differ, complicating fair comparison.

6.2.2. Evaluation Framework Recommendations

More comprehensive evaluation requires:

- **Standardized Protocols**: Development of standardized evaluation protocols specifying prompting strategies, sampling parameters, and scoring methodologies.
- **Dynamic Benchmarks**: Creation of continuously updated benchmarks with fresh problems to minimize contamination concerns.
- **Composite Metrics**: Use of composite metrics that evaluate multiple dimensions including correctness, efficiency, reliability, and safety.
- **Real-World Task Evaluation**: Emphasis on evaluation in realistic task settings that capture the complexity of practical applications.
- **Ablation Studies**: Systematic ablation studies to understand the contribution of different architectural components and training techniques.

6.3. Limitations and Challenges

Despite significant advances, several limitations persist:

1. **Evaluation Consistency**: Lack of standardized evaluation protocols and inconsistent reporting of results hampers meaningful comparison across models [2].
2. **Specialized Domain Performance**: While general capabilities improve, specialized domains often require fine-tuning, domain-specific training data, or custom approaches to achieve acceptable performance levels.
3. **Safety and Alignment**: Ensuring consistent alignment with human values across diverse contexts remains challenging. Models can exhibit unexpected behaviors in novel situations or adversarial scenarios.
4. **Computational Requirements**: Despite efficiency improvements through MoE and other techniques, advanced models remain computationally intensive. Training costs and inference requirements limit accessibility.
5. **Environmental Impact**: Energy consumption and carbon footprint of training and inference operations raise sustainability concerns. Continued efficiency improvements are necessary to address environmental impact.
6. **Interpretability**: Understanding model decision-making processes remains difficult. Limited interpretability complicates debugging, safety verification, and regulatory compliance.
7. **Factual Accuracy**: Models can generate plausible but incorrect information. Ensuring factual accuracy, particularly in high-stakes domains, requires additional verification mechanisms.
8. **Contextual Understanding**: While context windows extend to 1M tokens, effective utilization of such extensive context varies. Models may struggle to maintain coherence and utilize information from distant context.

 doi:10.20944/preprints202602.0537.v1

6.4. Emerging Trends and Future Directions

Several trends are likely to shape LLM development in the near future:

- **Specialized Models**: Development of domain-specific models optimized for particular applications. While general-purpose models continue to improve, specialized models may offer superior performance in focused domains.
- **Efficiency Improvements**: Continued refinement of MoE and other efficient architectures. Novel approaches including dynamic neural architectures and adaptive computation may further improve efficiency.
- **Multimodal Integration**: Deeper fusion of text, vision, audio, and structured data processing. Future models will likely process diverse input modalities more seamlessly.
- **Reasoning Enhancements**: Improved capabilities for complex, multi-step reasoning tasks. Explicit reasoning frameworks and verification mechanisms will enhance reliability in challenging domains.
- **Democratization**: Increased accessibility through open-weight models, reduced computational requirements, and improved deployment tools. Lowering barriers to entry will expand the developer ecosystem.
- **Tool Integration**: More sophisticated integration with external tools and systems. Agentic capabilities will expand as tool-use frameworks mature.
- **Personalization**: Development of techniques for efficient personalization and adaptation to individual users or specific use cases without extensive retraining.
- **Continuous Learning**: Mechanisms for continuous learning and knowledge updating that enable models to stay current without complete retraining.

The rapid pace of innovation suggests that current architectural approaches will continue to evolve, potentially giving rise to new paradigms that further enhance capabilities while addressing current limitations.

Declaration of Review Nature

This work is exclusively a survey paper synthesizing existing published research. No novel experiments, data collection, or original algorithms were conducted or developed by the authors. All content, including findings, results, performance metrics, architectural diagrams, and technical specifications, is derived from and attributed to the cited prior literature. The authors' contribution is limited to the compilation, organization, and presentation of pre-existing public knowledge. Any analysis or commentary is based solely on the information contained within the cited works. Figures and tables are visual representations of data and concepts described in the referenced sources.

Declaration of AI Assistance

Artificial intelligence tools, including Anthropic AI and DeepSeek, were used to assist in merging, organizing, and structuring portions of this manuscript. These tools were employed solely for editorial and organizational support. All technical content, interpretations, and conclusions remain the responsibility of the cited authors in this paper.

7. Conclusions

This comprehensive analysis of next-generation large language models reveals a rapidly evolving landscape characterized by architectural innovation, strategic divergence, and expanding application domains. The expansion of context windows to 1 million tokens, refinement of Mixture-of-Experts architectures, and deepening multimodal integration represent significant advances that enhance both capabilities and efficiency.

Claude Opus 4.6 represents a significant advancement in large language model architecture and performance, establishing new standards in extended context processing, reasoning capabilities, and practical utility. Through its 1 million token context window, refined Mixture-of-Experts architecture,

 doi:10.20944/preprints202602.0537.v1

and advanced tool-use framework, Opus 4.6 demonstrates the continued evolution of LLM technology beyond simple scaling toward more sophisticated architectural innovations.

Our analysis reveals several key insights:

- **Architectural Innovation**: Extended context windows, exemplified by Opus 4.6's 1M token capability, enable new classes of applications requiring deep comprehension of lengthy documents and complex reasoning chains. MoE architectures provide a viable path to maintaining or improving performance while significantly reducing computational requirements.
- **Strategic Divergence**: Models exhibit distinct strengths aligned with their architectural choices and training objectives, creating a fragmented but complementary ecosystem of solutions. Specialized capabilities in coding, tool use, and domain-specific reasoning reflect strategic architectural choices that differentiate competing models.
- **Practical Applications**: Real-world applications demonstrate practical value across software development, healthcare, regulatory compliance, and other domains. However, performance evaluation across diverse domains shows that specialized applications often require tailored approaches.
- **Persistent Challenges**: Challenges in evaluation consistency, specialized domain performance, safety, and alignment require continued research attention. Architectural trade-offs involve complex balancing of performance, efficiency, cost, and implementation considerations.

The competitive landscape characterized by rapid innovation and strategic divergence suggests continued architectural evolution in LLM development. Claude Opus 4.6's architectural choices—particularly its emphasis on extended context and reasoning depth—represent one strategic direction in this evolving landscape, offering significant advantages for applications requiring deep comprehension and complex problem-solving.

Future research should focus on several areas:

1. **Evaluation Frameworks**: Development of more comprehensive evaluation frameworks that capture real-world performance across diverse applications and enable meaningful comparisons across models.
2. **Efficiency Innovations**: Investigation of architectural innovations that further improve efficiency while maintaining or enhancing capabilities, addressing both computational cost and environmental impact.
3. **Specialized Architectures**: Exploration of specialized architectures optimized for specific domains and applications, balancing the benefits of specialization against the versatility of general-purpose models.
4. **Safety and Alignment**: Analysis of long-term implications of architectural choices on safety, alignment, and societal impact. Development of more robust safety mechanisms and alignment techniques.
5. **Standardized Benchmarking**: Development of standardized benchmarking methodologies with consistent protocols, dynamic problem sets, and real-world task evaluation.
6. **Domain Adaptation**: Improved specialized domain performance through targeted training approaches, efficient fine-tuning techniques, and domain-specific architectural adaptations.

As LLM technology continues to advance, architectural analysis and performance benchmarking will remain essential for understanding capabilities, identifying limitations, and guiding future development. The rapid pace of innovation in this field suggests that current capabilities represent only an intermediate stage in the evolution of artificial intelligence systems. Continued research and development, informed by rigorous evaluation and responsible deployment practices, will drive further advances in LLM technology and its applications across society. Claude Opus 4.6 represents an important milestone in this ongoing evolution, demonstrating the potential of architectural innovation to enable new capabilities and applications.

 doi:10.20944/preprints202602.0537.v1

References

1. A. Valiulla. Comparative Analysis of Next-Generation Large Language Models (LLMs): Architectural Advances, Reasoning Capabilities, and Multimodal Integration (2024-2025). [Online]. Available: https://papers.ssrn.com/abstract=5329049
2. D. D. Meva and H. Kukadiya, "Performance Evaluation of Large Language Models: A Comprehensive Review," vol. 12, pp. 109–114.
3. Anthropic's Claude Opus 4.6 Debuts with 1M Token Context Window. [Online]. Available: https://winbuzzer.com/2026/02/05/anthropic-claude-opus-46-1m-token-context-xcxwbn/
4. Claude Opus 4.6. [Online]. Available: https://www.anthropic.com/news/claude-opus-4-6
5. What's new in Claude 4.6. Claude API Docs. [Online]. Available: https://platform.claude.com/docs/en/about-claude/models/whats-new-claude-4-6
6. A. Morgan. AI War: 20 Minutes After Claude Opus 4.6, OpenAI Strikes Back With GPT-5.3 Codex. Ucstrategies News. [Online]. Available: https://ucstrategies.com/news/ai-war-20-minutes-after-claude-opus-4-6-openai-strikes-back-with-gpt-5-3-codex/
7. Zoer. GLM-4.6 vs Claude Sonnet 4.5: Which AI Model Wins in 2025? Zoer's Blog. [Online]. Available: https://zoer.ai/posts/zoer/glm-4-6-vs-claude-sonnet-4-5-comparison-661
8. Context Curves Behavior: Measuring AI Relational Dynamics with RCI[v2] | Preprints.org. [Online]. Available: https://www.preprints.org/manuscript/202601.1881
9. Building a C compiler with a team of parallel Claudes. [Online]. Available: https://www.anthropic.com/engineering/building-c-compiler
10. Zoer. GLM-4.6 vs Claude Sonnet 4.5: Performance Benchmark 2025. Zoer's Blog. [Online]. Available: https://zoer.ai/posts/zoer/glm-4-6-vs-sonnet-4-5-benchmark
11. A. Katranji, A. D. Vries, A. Katranji, and M. Zalzaleh. Comparative Accuracy of Large Language Models for CPT Coding Assignments from Surgical Procedure Notes. [Online]. Available: https://www.researchsquare.com/article/rs-8475390/v1
12. C. A. L. Flores, A. T. Soto, M. D. T. Soto, and F. S. Reyes, "Automated Information Extraction from Construction Regulations Using LangChain: A Case Study in Aguascalientes," in *Artificial Intelligence – COMIA 2025*, L. Martínez-Villaseñor, B. Martínez-Seis, and O. Pichardo, Eds. Springer Nature Switzerland, pp. 42–53.
13. Claude Opus 3 vs. GLM-4.6 Comparison. [Online]. Available: https://sourceforge.net/software/compare/Claude-3-Opus-vs-GLM-4.6/
14. Claude Opus 4.5 vs GLM 4.6 | LLM Comparison. Agentset. [Online]. Available: https://agentset.ai/llms/compare/claude-opus-45-vs-glm-46
15. Claude 4 Opus vs. Gemini 2.5 pro vs. OpenAI o3: Coding comparison - Composio. [Online]. Available: https://composio.dev/blog/claude-4-opus-vs-gemini-2-5-pro-vs-openai-o3
16. Claude Opus 4.5 Benchmarks and Analysis. [Online]. Available: https://artificialanalysis.ai/articles/claude-opus-4-5-benchmarks-and-analysis
17. F.-F. Zhao, H.-J. He, J.-J. Liang, J. Cen, Y. Wang, H. Lin, F. Chen, T.-P. Li, J.-F. Yang, L. Chen, and L.-P. Cen, "Benchmarking the performance of large language models in uveitis: A comparative analysis of ChatGPT-3.5, ChatGPT-4.0, Google Gemini, and Anthropic Claude3," vol. 39, no. 6, pp. 1132–1137. [Online]. Available: https://www.nature.com/articles/s41433-024-03545-9
18. A. Banerjee and I. Lavie, "Large Language Model Fingerprints From Normal Interaction."
19. Smart Contracts \ red.anthropic.com. [Online]. Available: https://red.anthropic.com/2025/smart-contracts/