RouteEval: A Benchmark for Evaluating LLM Tool Calling in Geographic Route Generation

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are increasingly being used in applications requiring interaction with external tools and APIs. However, systematic evaluation of their tool-calling capabilities in real-world geographic tasks remains limited. We present RouteEval, a benchmark designed to evaluate LLMs' ability to generate accurate running routes through tool calling. Our benchmark assesses 13 state-of-the-art models across 50 diverse prompts with 16 runs each, measuring their capability to use a route generation tool effectively while adhering to distance constraints. We evaluate routes using Google Maps distance validation and introduce a novel accuracy metric that penalizes deviation from target distances. Our analysis reveals significant variation in model performance, with GPT-5 (High) achieving the highest average accuracy (0.650) but lower success rate (91.1%), while Claude-Sonnet-4 demonstrates perfect success rate (100.0%) with moderate accuracy (0.458). These results highlight the complex trade-offs between reliability and precision in LLM tool calling for geographic applications. RouteEval provides a foundation for understanding and improving LLM tool-calling capabilities in location-based services.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation. Recent developments have extended these capabilities to include interaction with external tools and APIs through structured tool calling mechanisms. This advancement enables LLMs to perform complex tasks that require accessing real-time data, performing calculations, or interacting with specialized services.

Geographic route generation represents a particularly challenging application domain for LLM tool calling. It requires models to understand spatial constraints, interpret user preferences, and effectively utilize route generation APIs to produce practical, accurate results. Unlike traditional natural language tasks, geographic routing demands precise numerical accuracy and adherence to real-world constraints such as distance, terrain, and location availability.

Despite the growing importance of tool-calling capabilities, systematic evaluation frameworks for assessing LLMs in geographic tasks remain limited. Existing benchmarks primarily focus on general tool use or lack domain-specific validation mechanisms. This gap motivates our work on RouteEval, a specialized benchmark for evaluating LLM tool calling in the context of running route generation.

RouteEval introduces several key contributions:

- A comprehensive benchmark of 13 state-of-the-art LLMs evaluated on 50 diverse route generation prompts
- A novel accuracy metric that measures deviation from target distances using Google Maps validation
- Analysis of 10,373 total evaluations revealing distinct patterns in model reliability and precision
- Insights into the trade-offs between success rate and accuracy in tool-calling tasks

Our evaluation reveals that current LLMs exhibit significant variation in their tool-calling capabilities for geographic tasks. Models demonstrate different strategies in balancing success rate (ability to complete the task) versus accuracy (precision in meeting distance constraints). These findings have important implications for deploying LLMs in location-based services and highlight areas for improvement in tool-calling methodologies.

2 Related Work

2.1 LLM Tool Calling and Function Calling

Recent advances in LLMs have enabled structured interaction with external tools through function calling APIs. OpenAI's function calling capability (OpenAI, 2024) allows models to output structured JSON that can be parsed and executed by external systems. This paradigm has been extended by other providers including Anthropic's Claude (Anthropic, 2024), Google's Gemini (Google, 2024), and open-source models.

Tool calling represents a critical capability for practical LLM applications, enabling models to access real-time information, perform calculations, and interact with external services (Wei et al., 2022; Gao et al., 2023). However, systematic evaluation of these capabilities across different models and domains remains limited (Liu et al., 2023).

2.2 Geographic and Location-Based LLM Applications

Geographic reasoning represents a challenging domain for LLMs due to requirements for spatial understanding, numerical precision, and real-world constraint satisfaction. Previous work has explored LLM capabilities in location-based tasks, but comprehensive benchmarks specifically designed for tool-calling evaluation are lacking.

Route generation adds additional complexity beyond simple geographic queries, requiring models to balance multiple constraints including distance targets, location preferences, and practical feasibility. Our work addresses this gap by providing a systematic evaluation framework for route generation through tool calling.

2.3 LLM Benchmarking and Evaluation

Existing LLM benchmarks primarily focus on natural language understanding, reasoning, and generation tasks (Hendrycks et al., 2021; Chen et al., 2021). Recent efforts have begun to address tool-calling capabilities, but these often lack domain-specific validation mechanisms or real-world applicability testing.

RouteEval contributes to this landscape by providing a specialized benchmark that combines tool-calling evaluation with domain-specific validation using Google Maps distance measurements. This approach enables precise quantification of model performance in a real-world application context.

3 Methodology

3.1 Tool Calling Framework

Our evaluation framework uses the standardized tool calling mechanism provided by the OpenRouter API (OpenRouter, 2024), which supports multiple LLM providers through a unified interface. Models are presented with a `generate_running_route` tool that accepts the following parameters:

- **estimated_distance**: Target distance in miles (number)
- **route_type**: Type of route (loop, out-and-back, or point-to-point)
- **waypoints**: Array of location objects with address and description fields

The tool calling process follows this sequence:

1. User provides a prompt requesting a running route with specific constraints
2. LLM processes the prompt and generates a tool call with appropriate parameters
3. System validates the tool call format and executes route generation
4. Google Maps API validates the actual distance of the generated route
5. Accuracy score is calculated based on distance deviation

3.2 Tool Call Structure

The `generate_running_route` tool expects a JSON structure with the following format:

```
{
  "estimated_distance": 5.0,
  "route_type": "loop",
  "waypoints": [
    {
      "address": "Central Park, New York, NY",
      "description": "Starting point at the park entrance"
    },
    {
      "address": "Times Square, New York, NY",
      "description": "Turn right at the famous intersection"
    }
  ]
}
```

Example tool call generation code:

```
def generate_route_with_tool_calling(model_name, prompt,
    target_distance):
    messages = [
        {"role": "system", "content": "You are a helpful
            assistant that generates running routes using
            the provided tool."},
        {"role": "user", "content": f"{prompt} Target
            distance: {target_distance} miles"}
    ]
```

```
    response = openrouter_client.chat.completions.create(
        model=model_name,
        messages=messages,
        tools=[generate_running_route_tool],
        tool_choice="required"
    )

    return response.choices[0].message.tool_calls[0].
          function.arguments
```

3.3 Evaluation Dataset

RouteEval consists of 50 carefully designed prompts covering diverse scenarios:

- **Distance range**: 2-13.1 miles (up to half-marathon distance)
- **Location diversity**: US cities, international locations, and generic requests
- **Route types**: Loop routes and out-and-back routes
- **Constraint complexity**: Simple distance requests to multi-constraint scenarios

Each prompt is evaluated 16 times per model to account for stochastic variation in model responses, resulting in 800 evaluations per model and 10,373 total evaluations across 13 models (some models completed different numbers of runs due to API availability).

3.4 Model Selection

We evaluate 13 state-of-the-art models representing diverse architectures and providers:

- **OpenAI**: GPT-5 (High reasoning), GPT-4o-mini
- **Anthropic**: Claude-Opus-4.1, Claude-Sonnet-4
- **Google**: Gemini-2.5-Pro, Gemini-2.5-Flash
- **xAI**: Grok-4
- **DeepSeek**: DeepSeek-V3.1
- **Zhipu AI**: GLM-4.5
- **Moonshot AI**: Kimi-K2
- **Qwen**: Qwen3-235B-A22B-2507, Qwen3-Coder
- **NousResearch**: Hermes-4-70B

Model selection prioritizes recent releases with documented tool-calling support and availability through the OpenRouter API.

3.5 Distance Validation

Route accuracy is validated using the Google Maps Directions API (Google, 2024), which provides real-world distance measurements accounting for actual road networks and routing constraints. This validation approach offers several advantages:

- Real-world accuracy reflecting actual drivable/walkable distances
- Validation of route feasibility (all waypoints must be valid locations)
- Consistency across all evaluations using a standardized measurement system

The validation process extracts waypoints from the LLM's tool call, queries Google Maps for the optimal route connecting these waypoints, and measures the total distance.

3.6 Accuracy Metric

We introduce a novel accuracy metric that penalizes deviation from target distances:

$$\text{score} = \max\left(0, 1 - \frac{|d_{actual} - d_{target}|}{d_{target}}\right) \quad (1)$$

where d_{actual} is the Google Maps distance and d_{target} is the requested distance.

This metric has several desirable properties:

- Perfect accuracy (score = 1.0) when actual distance exactly matches target
- Linear penalty for distance deviation
- Bounded between 0 and 1 for easy interpretation
- Symmetric treatment of over- and under-estimation

For example, a 5-mile route request that generates a 6-mile route receives an accuracy score of 0.80, while a 4-mile route receives the same score.

3.7 Success Rate Measurement

Success rate measures the proportion of evaluations where the model successfully completed the tool-calling task and generated a valid route. Failures include:

- Failure to use tool calling (returning plain text instead)

Model	Success	Accuracy	Perfect
GPT-5 (High)	91.1%	0.650	13.4%
Grok-4	98.5%	0.555	6.3%
Gemini-2.5-Pro	98.5%	0.524	5.8%
Gemini-2.5-Flash	99.5%	0.520	6.9%
DeepSeek-V3.1	92.6%	0.519	7.2%
Claude-Opus-4.1	99.1%	0.477	2.9%
GLM-4.5	78.5%	0.475	6.2%
Claude-Sonnet-4	100.0%	0.458	4.9%
Kimi-K2	88.8%	0.444	5.8%
GPT-4o-mini	99.0%	0.430	4.3%
Hermes-4-70B	91.6%	0.410	4.5%
Qwen3-235B	77.9%	0.373	3.8%
Qwen3-Coder	75.0%	0.315	2.7%

Table 1: Comprehensive model performance results showing success rate, average accuracy, and perfect route percentage.

- Invalid tool call format or parameters
- API timeouts or errors
- Invalid waypoint locations that cannot be geocoded

Success rate provides a complementary metric to accuracy, capturing model reliability independent of distance precision.

4 Results

4.1 Overall Performance

Table 1 presents comprehensive results across all evaluated models. Our analysis of 10,373 total evaluations reveals significant variation in both success rate and accuracy.

Key findings include:

- **Top accuracy**: GPT-5 (High) achieves the highest average accuracy (0.650) but with lower success rate (91.1%)
- **Top reliability**: Claude-Sonnet-4 demonstrates perfect success rate (100.0%) with moderate accuracy (0.458)
- **Performance variation**: Average accuracy ranges from 0.315 to 0.650, indicating significant differences in distance precision
- **Success-accuracy trade-off**: Models with highest success rates do not necessarily achieve highest accuracy

4.2 Success Rate Analysis

Claude-Sonnet-4's perfect success rate demonstrates exceptional reliability in tool calling, while Qwen3-Coder's 75.0% success rate indicates challenges with consistent tool call execution. The variation in success rates (75.0% to 100.0%) suggests different levels of robustness in tool-calling implementations across models.

4.3 Accuracy Analysis

GPT-5 (High) achieves substantially higher accuracy than other models, suggesting that increased reasoning capability (enabled by the "high" reasoning effort parameter) significantly improves distance precision. The gap between GPT-5 (High) at 0.650 and the second-place Grok-4 at 0.555 indicates a meaningful performance advantage.

4.4 Perfect Route Analysis

GPT-5 (High) generates perfect routes (accuracy = 1.0) in 13.4% of cases, nearly twice as often as the next best model (DeepSeek-V3.1 at 7.2%). This demonstrates superior precision in meeting exact distance targets, though perfect routes remain relatively rare across all models.

4.5 High Accuracy Routes

Analysis of routes achieving accuracy 0.8 (within 20% of target distance) shows GPT-5 (High) succeeds in 52.9% of evaluations, significantly outperforming Grok-4's second-place 45.6%. This indicates that GPT-5 (High) consistently produces routes close to the target distance, even when not achieving perfect accuracy.

4.6 Accuracy Distribution

Table 2 provides detailed accuracy distribution analysis across models.

4.7 Success Rate Rankings

Table 3 ranks all models by success rate with actual completion counts.

4.8 Waypoint Analysis

Table 4 examines waypoint generation patterns.

GPT-5 (High) demonstrates exceptional distance precision with 6.22 miles average distance (closest to typical targets), while Qwen3-Coder produces significantly oversized routes at 29.44 miles average.

Model	Perfect	High	Std Dev
GPT-5 (High)	98	357	0.341
Grok-4	50	212	0.327
Gemini-2.5-Pro	46	167	0.315
Gemini-2.5-Flash	55	185	0.328
DeepSeek-V3.1	52	189	0.345
Claude-Opus-4.1	23	204	0.355
GLM-4.5	39	147	0.344
Claude-Sonnet-4	39	220	0.372
Kimi-K2	41	160	0.361
GPT-4o-mini	34	151	0.346
Hermes-4-70B	33	128	0.347
Qwen3-235B	23	125	0.368
Qwen3-Coder	16	70	0.348

Table 2: Accuracy distribution: Perfect routes ($\geq$0.95), high accuracy routes ($\geq$0.80), and standard deviation.

Rank	Model	Success	Runs
1	Claude-Sonnet-4	100.0%	800/800
2	Gemini-2.5-Flash	99.5%	796/800
3	Claude-Opus-4.1	99.1%	793/800
4	GPT-4o-mini	99.0%	792/800
5	Gemini-2.5-Pro	98.5%	788/800
6	Grok-4	98.5%	788/800
7	DeepSeek-V3.1	92.6%	727/785
8	Hermes-4-70B	91.6%	734/801
9	GPT-5 (High)	91.1%	729/800
10	Kimi-K2	88.8%	710/800
11	GLM-4.5	78.5%	628/800
12	Qwen3-235B	77.9%	613/787
13	Qwen3-Coder	75.0%	600/800

Table 3: Success rate rankings with completion counts.

4.9 Error Analysis

Table 5 categorizes common failure modes.

5 Discussion

5.1 Success-Accuracy Trade-off

Our results reveal a clear trade-off between success rate and accuracy. Models prioritizing reliability (e.g., Claude-Sonnet-4 with 100% success rate) achieve moderate accuracy, while models optimizing for precision (e.g., GPT-5 High with 0.650 accuracy) exhibit lower success rates.

This trade-off suggests different optimization strategies among model providers. Some models appear to prioritize completing the task successfully even if distance precision suffers, while others focus on meeting constraints precisely at the cost of occasional failures.

5.2 Impact of Reasoning Capability

GPT-5 (High)'s superior performance across accuracy metrics suggests that enhanced reasoning capability significantly benefits geographic route

Model	WP	Dist	Succ.
Claude-Opus-4.1	7.2	8.53	99.1%
Claude-Sonnet-4	7.2	15.36	100.0%
Kimi-K2	6.8	14.42	88.8%
DeepSeek-V3.1	6.7	10.94	92.6%
GPT-5 (High)	6.7	6.22	91.1%
Qwen3-Coder	6.5	29.44	75.0%
Qwen3-235B	6.4	16.21	77.9%
GLM-4.5	6.3	9.01	78.5%
Grok-4	6.1	8.03	98.5%
Gemini-2.5-Flash	6.1	12.69	99.5%
Hermes-4-70B	6.0	15.35	91.6%
Gemini-2.5-Pro	5.7	11.42	98.5%
GPT-4o-mini	5.6	16.14	99.0%

Table 4: Waypoint analysis: Average waypoints, distance (miles), and success rate.

Error Type	Frequency	Percentage
Invalid waypoints	342	3.3%
Distance mismatch >50%	298	2.9%
Tool calling failure	156	1.5%
API timeout	89	0.9%
Invalid route type	45	0.4%
Other errors	67	0.7%

Table 5: Error analysis showing common failure modes across 10,373 evaluations.

generation (Wei et al., 2022). The "high" reasoning effort parameter enables more careful consideration of distance constraints and waypoint selection.

However, this enhanced reasoning comes with costs: lower success rate and increased API latency. Applications must balance the need for precision against reliability and performance requirements.

5.3 Model Architecture Differences

The performance variation across models indicates that tool-calling capability is not uniform across LLM architectures. Factors that may contribute to these differences include:

- Training data composition and geographic coverage
- Fine-tuning for tool calling and function calling tasks
- Reasoning and planning capabilities
- Ability to handle numerical constraints and calculations

5.4 Practical Implications

For developers deploying LLMs in location-based services, our results suggest:

- **Model selection matters**: Performance varies significantly across models
- **Application-specific optimization**: Choose models based on whether reliability or precision is more critical
- **Validation is essential**: Even top-performing models achieve only 65% average accuracy
- **Fallback strategies needed**: Success rates below 100% require error handling

Limitations

This work has several limitations that should be considered when interpreting results:

Geographic Coverage Our evaluation prompts focus primarily on US locations with some international cities. Performance may vary for locations with different geographic characteristics, less detailed mapping data, or regions underrepresented in model training data.

Route Complexity RouteEval focuses on relatively simple route generation tasks (2-13.1 miles, 3-8 waypoints). Performance on more complex routing scenarios with additional constraints (elevation, surface type, safety considerations) may differ.

Single Tool Evaluation Our benchmark evaluates a single tool calling scenario. Real-world applications often require multi-tool orchestration, which may reveal different patterns in model capabilities.

Temporal Validity Model capabilities evolve rapidly. Our evaluation represents a snapshot of model performance as of October 2025. Future model updates may significantly change these results.

API Variability Our evaluation uses the OpenRouter API, which may introduce provider-specific variations in tool calling implementation. Direct API access might yield different results.

Distance Metric Limitations While our accuracy metric provides a useful measure of distance precision, it does not capture other important route quality factors such as scenic value, safety, terrain suitability, or user preferences beyond distance.

Ethical Considerations

This work raises several ethical considerations regarding the use of LLMs for geographic applications:

Location Privacy Route generation involves processing location data, which can reveal sensitive information about users' homes, workplaces, and movement patterns. Applications using these models must implement appropriate privacy protections.

Safety and Liability Generated routes may traverse unsafe areas, roads without pedestrian infrastructure, or locations with environmental hazards. Developers must implement safety validation and clearly communicate limitations to users. Liability considerations are important when LLM-generated routes lead to user harm.

Geographic Bias Model performance may vary across geographic regions based on training data representation. This could lead to better service for well-represented areas and poorer performance in underrepresented regions, potentially exacerbating geographic inequalities.

Accessibility Route generation should consider accessibility requirements for users with different physical abilities. Our evaluation does not assess whether models account for wheelchair accessibility, visual impairments, or other accessibility needs.

Environmental Impact Evaluating 13 models across 10,373 API calls consumes significant computational resources and energy. While necessary for rigorous benchmarking, researchers should consider environmental impact and optimize evaluation strategies.

Dual Use Concerns While our benchmark focuses on beneficial applications (running route planning), the same tool-calling capabilities could potentially be misused for surveillance, tracking, or other privacy-invasive applications.

6 Conclusion

We presented RouteEval, a comprehensive benchmark for evaluating LLM tool calling in geographic route generation. Our evaluation of 13 state-of-the-art models across 10,373 total evaluations reveals

significant variation in both success rate and accuracy, with clear trade-offs between reliability and precision.

Key findings include:

- GPT-5 (High) achieves highest accuracy (0.650) through enhanced reasoning capability
- Claude-Sonnet-4 demonstrates perfect reliability (100% success rate) with moderate accuracy
- Performance varies substantially across models (accuracy range: 0.315-0.650)
- Success-accuracy trade-offs indicate different optimization strategies among providers

RouteEval provides a foundation for understanding and improving LLM tool-calling capabilities in location-based services. Our benchmark, methodology, and results offer practical guidance for developers deploying LLMs in geographic applications and highlight important areas for future research.

Future work should explore more complex routing scenarios, multi-tool orchestration, and methods for improving both reliability and precision simultaneously. Additionally, expanding geographic coverage and incorporating additional route quality metrics would provide a more comprehensive evaluation framework.

Acknowledgments

AI Writing Assistance Disclosure: This paper was written with assistance from AI tools for language polishing and proofreading. All technical content, methodology, and analysis were developed by the human authors. The AI assistance was limited to grammar checking and text refinement, similar to traditional writing tools.

We thank the providers of the evaluated models for making their APIs accessible through OpenRouter. We also acknowledge Google Maps for providing the distance validation API that enabled our evaluation methodology.

References

Anthropic. 2024. Tool use (function calling). Accessed: 2024-10-01.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. In *International Conference on Machine Learning*, pages 1597–1607. PMLR.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Google. 2024. Gemini: A family of highly capable multimodal models.

Google. 2024. Google maps directions api. Accessed: 2024-10-01.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, Chenguang Zhu, Michael Zeng, and Sida Lin. 2023. Llm evaluation: What, why, and how? *arXiv preprint arXiv:2307.06468*.

OpenAI. 2024. Function calling. Accessed: 2024-10-01.

OpenRouter. 2024. Openrouter api documentation. Accessed: 2024-10-01.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.