In-Context Trajectory Learning for Construction Scheduling Using Large Language Models

Shanika Manamperi[1], Wei Peng[2] and Guomin Zhang[3]

[1,2,3]School of Engineering, RMIT University, Australia
s4061684@student.rmit.edu.au, wei.peng3@rmit.edu.au and kevin.zhang@rmit.edu.au

Abstract

Adoption of Artificial Intelligence (AI) is growing across the construction industry, but its use remains uneven. While AI is being increasingly adopted for information management workflows, its application to construction scheduling remains limited. It is a critical gap because construction scheduling is inherently dynamic and must respond to disruptions such as workforce constraints, procurement delays, coordination issues, and changing site conditions. Although traditional scheduling approaches have improved over time, they remain limited in supporting adaptive decision-making under uncertainty. To address this gap, this study proposes a trajectory-based learning approach for construction scheduling. A technical simulation environment was developed by modelling construction actions as trajectories. The proposed framework is demonstrated through a case study of ductwork installation in a six-story building. A dataset of 50 trajectories with 577 steps of thought-action-observation was constructed to capture diverse execution scenarios. An In-Context Learning (ICL) approach leveraging large language models (LLMs) was adopted to learn from training trajectories and predict subsequent construction actions. Zero-shot and few-shot prompting strategies were evaluated comparatively. The initial next-action prediction task was further extended with an additional experiment on predicting short sequences of future actions to better reflect real-world construction operations. Results show that Llama 3B achieved 76% accuracy in next-action prediction under the 5-shot setting, while GPT-5.4 demonstrated stronger reasoning capability in the extended experiment. This study highlights the potential of trajectory-based learning as an on-site decision-support mechanism and provides a preliminary foundation for future intelligent construction scheduling agents.

Keywords –

Construction; In-context learning; Large language models; Scheduling; Trajectory learning

1 Introduction

Despite technological advancements and applications in the industry, accurate construction scheduling remains a critical challenge. This is basically due to the inherent complexity and dynamic nature of construction work. In addition, other influencing factors such as labour, material and equipment availability, weather, change orders and economic fluctuations also contribute to unrealistic scheduling. Traditional scheduling approaches range from the Critical Path Method (CPM) to construction scheduling software like MS Project and Primavera which remain deterministic with poor reasoning capability [1]. Research has also focused on applying Artificial Intelligence (AI) and Machine Learning (ML) to construction scheduling in terms of predicting completion percentages and evaluating project performance [2]. Moving forward, current research has applied Large Language Models (LLMs) to construction scheduling in terms of capturing delay causes and automatic schedule generation [3]. Additionally, integration of digital technologies like Digital Twins (DTs) with LLMs can be seen in enhancing robots for dynamic task allocation in construction [4]. Some studies have developed LLM-based frameworks for construction with the integration of robotics by leveraging Industry Foundation Classes (IFC) [5]. These frameworks have been successful in guiding robots to perform tasks like brick-laying and 3D printing [5]. Despite these advances, existing approaches remain limited in their ability to handle construction uncertainties in an adaptive manner. Most AI and LLM-based scheduling approaches depend on predefined methods offering deterministic schedule generation instead of learning through sequential behaviours. This highlights a research gap in developing scheduling approaches that can reason under uncertainties and better address current industry needs. To address this gap, this study introduces an agentic approach for construction scheduling that learns sequencing behaviour from trajectories under uncertainty.

LLMs are transformative technologies with applications not only in construction but also in other sectors like finance and manufacturing [6]. In-context

learning (ICL) is a mechanism through which LLMs perform a task by learning from instructions and examples given directly in the prompt. ICL eliminates the need for complex fine-tuning which frequently acts as a bottleneck in existing ML approaches [7]. Hence, an ICL method was applied in this study leveraging LLMs, using training trajectory data as the context for the user prompt. This study demonstrates HVAC (Heating, Ventilation, and Air Conditioning) installation as a case study where both positive (successful) and negative (unsuccessful) trajectories were provided as in-context examples. Dynamic construction interactions which include task dependencies, rework and coordination between site personnel were incorporated to infer decision patterns. Ultimately, this study examines whether ICL enables LLMs to reason over construction scheduling trajectories.

2 Methodology

2.1 Trajectory data

This study models the interaction between the agent and the environment as trajectories which provides the structural representation used for learning. Trajectories capture the complete or partial execution of a construction task over time through a sequence of decisions and outcomes. In this study schedule related operations are represented as thought-action-observation (t-a-o) trajectories, where thought denotes what needs to be done, action denotes the managerial or operational decision taken, and observation denotes the resulting outcome (Figure 1) [8].

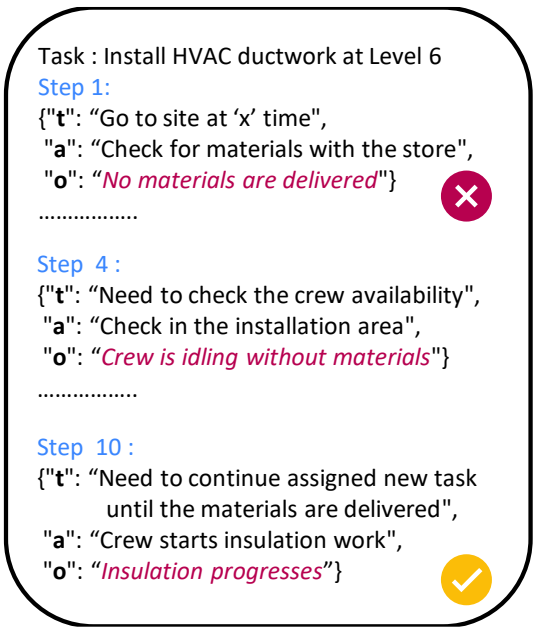

Figure 1. Trajectory representation

Dynamic scheduling factors such as crew availability, safety hazards, material delivery delays, and environmental conditions are embedded explicitly at the step level within the observation component. This enables the trajectories to capture real-world construction uncertainties and intermediate decision consequences. These trajectories are encoded into the task environment in a structured order, where the state space ($\mathcal{S}$) describes the context at a given step and the action space ($\mathcal{A}$).

This study used an open-source time-motion dataset which includes a long chain of events resulting from video footage [9]. It captures the activities of MEP (Mechanical, Electrical, and Plumbing) workers in a six-storey multifamily building in Finland which showed elevated levels of wasted efforts. A total of 50 trajectories were constructed for the HVAC installation task, including 12 positive trajectories and 38 negative trajectories. Positive trajectories contain correct execution steps throughout the full sequence. Negative trajectories include either trajectories with an unsuccessful final outcome, or trajectories with a successful final outcome but incorrect intermediate steps. Negative trajectories were included since the process of human learning not only involves observing successful executions but also includes experiencing failures [10].

2.2 ICL approach

ICL was adopted to evaluate LLMs for construction action prediction in schedule decision support. Instruction-finetuned open-source LLMs, namely Llama-3.2-3B-Instruct and Qwen 2.5-3B-Instruct were used in this study. These models were selected because they represent relatively lightweight and practically deployable LLMs suitable for early-stage evaluation of on-site decision-support scenarios. These models were accessed through a Hugging Face API token [11]. Training trajectories were given as contextual examples within the user prompt enabling models to learn scheduling patterns. Data were split based on the 80/20 rule. The held-out test trajectories served as the ground truth with the original t-a-o construction sequences.

During ICL, LLMs analysed unseen test trajectories and predicted the next action under uncertainty. However, this decision is solely based on the given trajectories of different scenarios. Zero-shot and few-shot prompting approaches were incorporated for comparative evaluation. In the zero-shot setting, the model receives no in-context trajectory examples, whereas in the few-shot setting, it learns from a small number of example trajectories provided in the prompt. Factors like the diversity and quality of examples shown in the in-context demonstrations may improve the quality of model output [7]. To assess the capability of ICL in distinguishing trajectory patterns, two conditions were studied: positive only and mixed positive-negative trajectory contexts.

Prompt was designed as using positive trajectory and (positive + negative) trajectory contexts to predict the next best action. Regarding example selection, few-shot examples were re-sampled for each run, and each model was evaluated across all test trajectories. The prompt was specified to generate only positive outcomes. For each condition, zero-shot, 1-shot, 5-shot and 10-shot prompting configurations were examined. Final reward was computed based on the Outcome Reward Model (ORM). ORM uses binary labels with "1" representing successful task completion and "0" for uncompleted tasks [12]. This study assigns a reward of "1" if the ground-truth label is "positive", whereas it assigns a value of "0" if the ground-truth label is "negative". Evaluation was conducted at the trajectory level by aggregating step-level rewards across each predicted trajectory and computing the average reward against the ground-truth trajectory. To evaluate whether the system can differentiate positive and negative trajectories, accuracy and a confusion matrix were computed, including true positives (TP) and true negatives (TN). Model outputs were evaluated over multiple runs, and accuracy was recorded. Subsequently, accuracy for each shot configuration was calculated by taking the average accuracy across all runs.

In real-world construction processes, scheduling decisions often involve not only the next action but also a short sequence of subsequent actions. To better reflect this practical need, the ICL setup for next-action prediction was extended with an additional experiment that evaluates the prediction of future action sequences with prior trajectory actions as input context. In addition

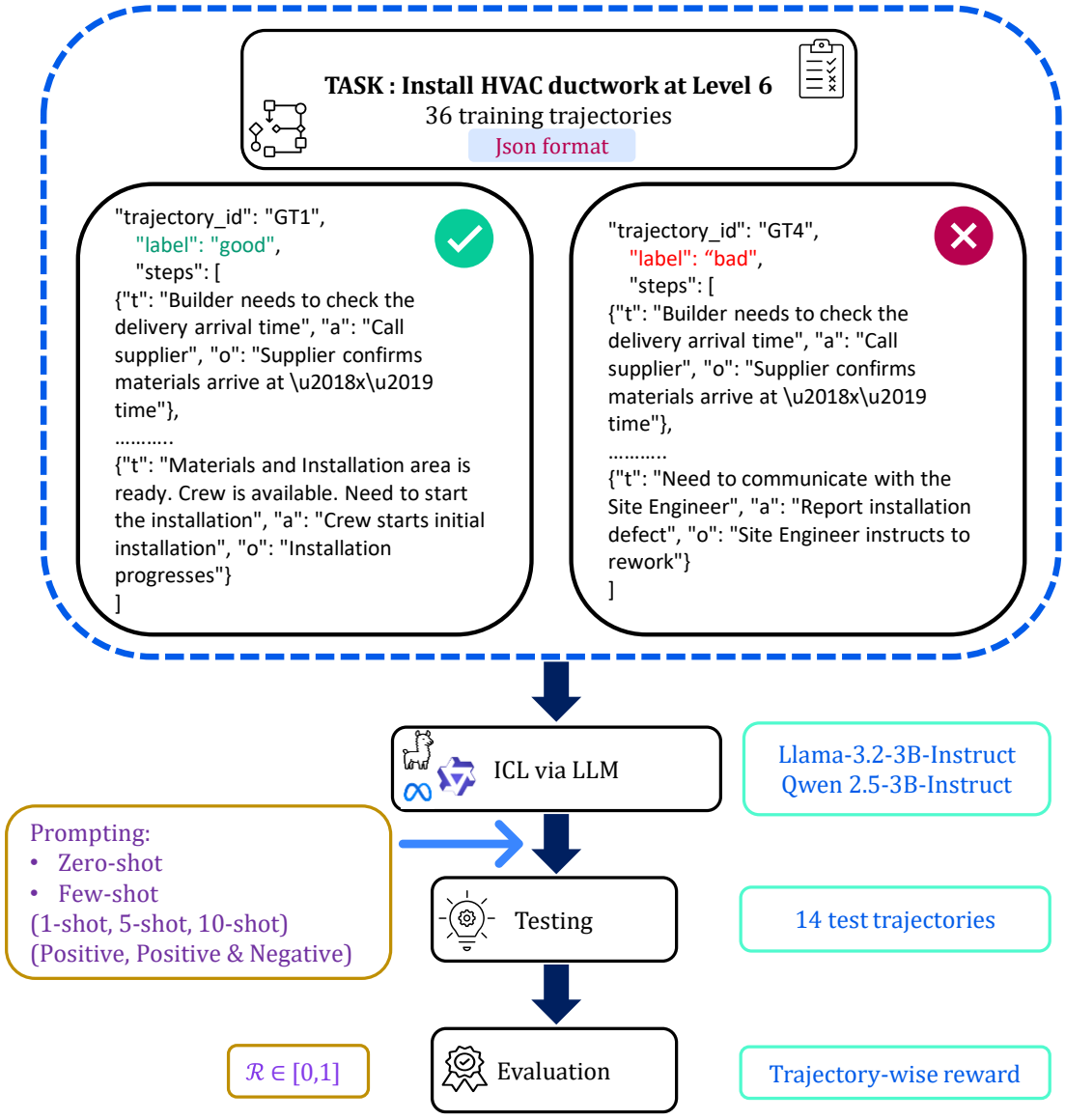

Figure 2. Methodology for preliminary ICL approach

to Llama and Qwen, this experiment included Mistral-7B-Instruct-v0.2, GPT-4o-mini, and GPT-5.4 to compare performance across both open-source and API-based models. Predicted trajectories were assessed manually.

3 Results and discussion

When the same unseen trajectory was tested with LLMs under positive and negative conditions the following outputs were obtained as indicated in Table 1.

Table 1. Accuracy levels of positive-negative trajectory context

Prompt	Qwen 2.5-3B-Instruct	Llama 3.2-3B-Instruct
Zero-shot	30%	40%
1-shot	76%	66%
5-shot	60%	76%
10-shot	58%	70%

Table 1 shows that neither model performed well under the zero-shot setting. Qwen achieved its best performance under the 1-shot configuration, with an accuracy of 76%, compared with 66% for Llama. In contrast, Llama performed better at the 5-shot and 10-shot settings, suggesting that it benefited more from additional in-context examples. However, both models showed lower performance at 10-shot than at 5-shot, indicating that adding more examples did not necessarily improve prediction accuracy. This suggests that excessive in-context examples may introduce less relevant or distracting context, reducing the models' ability to use the prompt effectively. Overall, Qwen performed best in the 1-shot setting, whereas Llama achieved its strongest performance in the 5-shot setting.

One possible reason for the variation in few-shot performance is output truncation. As the number of few-shot examples increases, the prompt becomes more complex and consumes more tokens during inference. Since each trajectory contains around 10 steps on average, larger few-shot settings increase both prompt length and response complexity. To manage this, generation was limited to 500 output tokens. As a result, the study mainly evaluated action prediction accuracy rather than the quality of the generated explanation. Example explanations are presented only for illustration, and their content may vary depending on the number and similarity of the in-context examples. An example from the 5-shot Llama setting is shown in Figure 3.

In addition to comparing different LLM architectures, an ablation study was conducted to examine the effect of model scale within the same model family by comparing Qwen 2.5-7B-Instruct and Qwen 2.5-3B-Instruct. As shown in Figure 4, the 7B model achieved higher accuracy than the 3B model in most settings, except for the 1-shot configuration. Zero-shot performance was nearly twice as high for the 7B model, suggesting stronger instruction-following capability. The 7B model also performed better under the 5-shot and 10-shot settings, indicating improved use of multiple in-context examples. These results suggest that increasing model scale may improve robustness to noisy context and strengthen the use of trajectory examples.

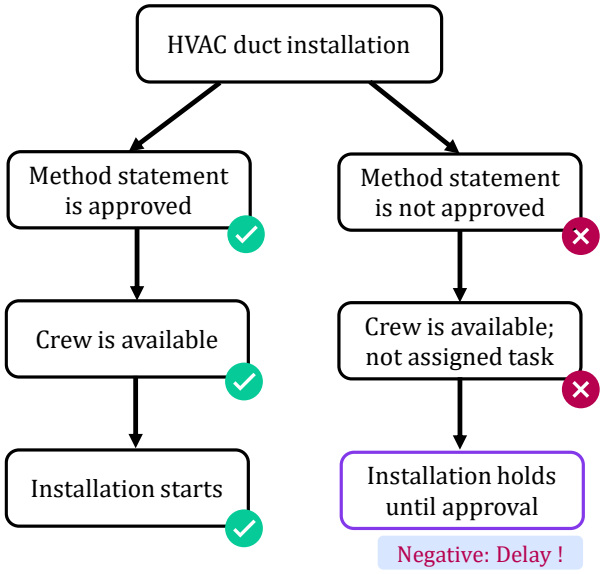

Figure 3. Next action prediction (5 shot Llama setting)

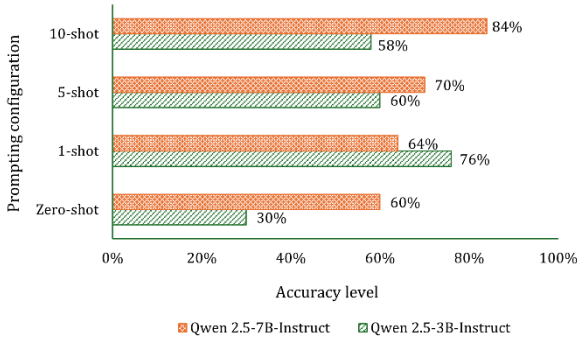

Figure 4. Accuracy levels of the ablation study

Figure 5 presents the cosine similarity values obtained in the extended experiment on predicting future action sequences across different LLMs and prompting configurations. Qwen and Llama showed patterns generally consistent with the next-action prediction results from the previous experiment, while Mistral performed at a similar level with slight variation across prompt settings. GPT-based models showed stronger performance than the open-source models, suggesting that trajectory-prediction performance may improve with model scale. Figure 6 shows an example future trajectory prediction generated by GPT-5.4.

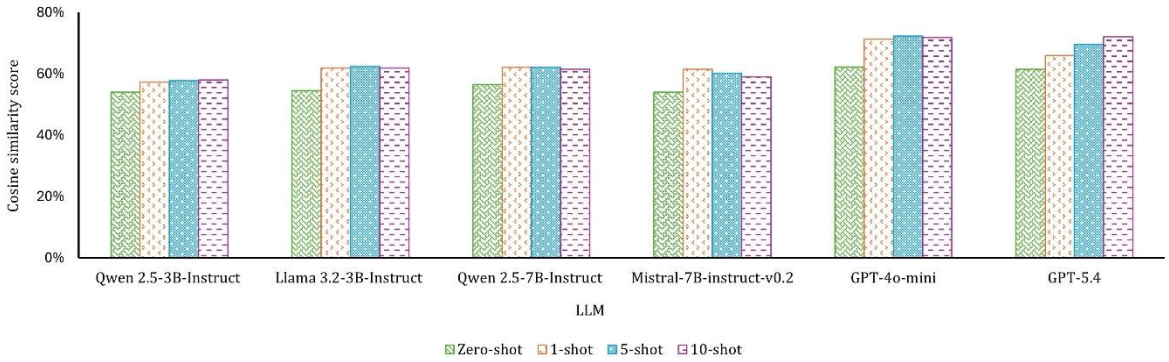

Figure 5. Cosine similarity scoring of different LLMs

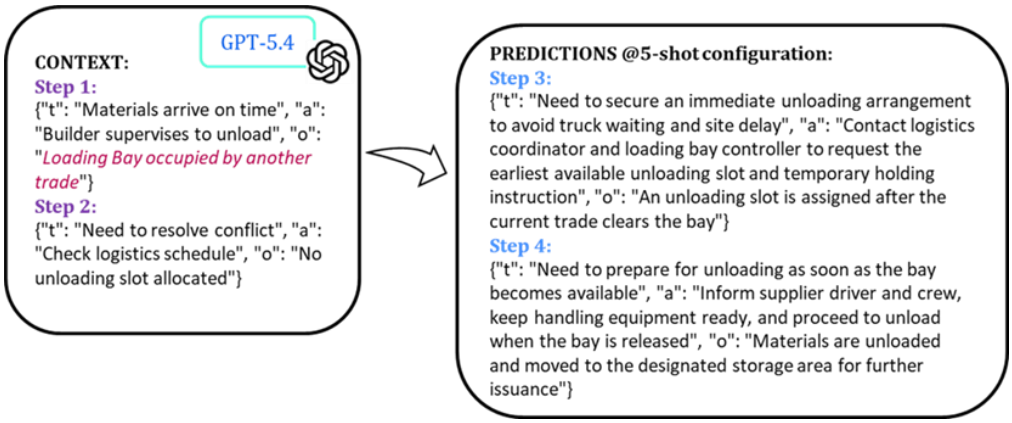

Figure 6. Prediction of trajectory using GPT-5.4

This study presents a trajectory-based ICL approach for construction scheduling. Rather than replacing conventional scheduling tools, the proposed method is designed as a decision-support mechanism that helps site teams interpret evolving situations and identify feasible follow-up actions. This work contributes to contrastive learning from positive and negative trajectory patterns, including intermediate-step failures that are often overlooked in conventional deterministic schedules.

One of the key limitations is the limited number of trajectories which limits the diversity of decision patterns available for unseen scenarios. Hence, it is recommended to train the model with a considerably bigger dataset, covering a wide range of cascading scenarios under uncertainty.

As an initial step towards developing a scheduling agent, this study provides a basis for future development. The next step is to advance this work towards an LLM-based scheduling agent by incorporating stepwise rewards and a mixed post-training method that includes supervised instruction fine-tuning and reinforcement learning. It would also be valuable to include larger scale LLM baseline models in future evaluations.

4 Conclusion

Timely project completion remains a persistent challenge in the construction industry adhering to its inherent complex nature and uncertainty. The proposed approach is designed to support adaptive scheduling decisions under disruptions by predicting future action sequences from prior trajectory context.

This study leads to the following key findings. First, the performance of ICL is influenced by both the number and quality of the in-context examples. Despite the small dataset, ICL appears to offer a practical alternative in settings where construction scheduling data are scarce. Additionally, Llama showed stronger performance in next-action prediction at the preliminary stage, while commercial LLMs such as GPT-5.4 performed better in the extended future-trajectory prediction experiment. This suggests that model architecture and scale may influence trajectory-learning performance.

This paper presents an exploratory study towards such a system, in which the agent learns construction sequences from both positive and negative trajectories and is evaluated on its ability to predict subsequent actions and short future trajectories. In practical terms, the contribution of the approach lies in its potential to support schedule adjustment under disruptions by recommending feasible next actions that reduce delay.

References

[1]. C. I. Lagos, R. F. Herrera, A. F. Mac Cawley, and L. F. Alarcón, "Predicting construction schedule performance with last planner system and machine learning," *Automation in Construction*, vol. 167, p. 105716, 2024/11/01, doi: https://doi.org/10.1016/j.autcon.2024.105716.

[2]. M.-Y. Cheng, Y.-H. Chang, and D. Korir, "Novel Approach to Estimating Schedule to Completion in Construction Projects Using Sequence and Non sequence Learning," *Journal of Construction Engineering and Management*, vol. 145, no. 11, p. 04019072, 2019, doi:https://doi.org/10.1061/(ASCE)CO.1943-7862.0001697.

[3]. A. Pal, D. Murguia, and C. Middleton, Automatic Inference of Construction Delays through Analysis of Weekly Progress Reports Using LLMs. 2024.

[4]. M. Deng, B. Fu, L. Li, and X. Wang, Integrating LLMs and Digital Twins for Adaptive Multi-Robot Task Allocation in Construction. 2025. doi: https://doi.org/10.48550/arXiv.2506.18178

[5]. S. Du, Y. Gao, W. Tong, and Y. Weng, "Towards Agent: LLM-based Framework for Robotic Construction by Leveraging IFC," in 4*2nd International Symposium on Automation and Robotics in Construction (ISARC 2025)*, pp. 358-368, Canada, 2025, doi: https://doi.org/10.22260/ISARC2025/0048.

[6]. M. Raiaan et al., "A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges," 2023. [Online]. Available: 10.36227/techrxiv.24171183. Preprints

[7]. Y. Emami, H. Zhou, S. Nabavirazavi, and L. Almeida, "LLM-Enabled In-Context Learning for Data Collection Scheduling in UAV-assisted Sensor Networks," 2025. [Online]. Available: https://doi.org/10.1109/JIOT.2025.3615410. Preprints

[8]. W. Xiong et al., Watch Every Step! LLM Agent Learning via Iterative Step-level Process Refinement. 2024, pp. 1556-1572. doi: https://arxiv.org/abs/2406.11176

[9]. C. Görsch, O. Seppänen, A. Peltokorpi, and R. Lavikka, "Unlocking Productivity: Revealing Waste and Hidden Disturbances Impacting MEP Workers," *Journal of Construction Engineering and Management*, vol. 150, no. 9, p. 04024108, 2024, https://doi.org/10.1061/JCEMD4.COENG-14204.

[10]. Y. Song, D. Yin, X. Yue, J. Huang, S. Li, and B. Y. Lin, "Trial and Error: Exploration-Based Trajectory Optimization for LLM Agents," in Annual Meeting of the Association for Computational Linguistics, 2024. doi: https://arxiv.org/abs/2403.02502

[11]. H. Face. "Hugging Face Hub Documentation." On-line. https://huggingface.co/docs, Accessed: 25/12/2025

[12]. P. Wang et al., "Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations," pp. 9426-9439, Bangkok, Thailand, August 2024: Association for Computational Linguistics, in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics* (Volume 1: Long Papers), doi: https://doi.org/10.18653/v1/2024.acl-long.510