Less is Better: A Comparative Analysis of Guidance Methods for Long-Context Window-Based Legal Interpretation

Jaekun Lee[1] **and Ghang Lee**[1,2*]

[1]Department of Architecture and Architectural Engineering, Yonsei University, Seoul, Republic of Korea
[2]Institute for Advanced Studies, Technical University of Munich, Munich, Germany
*Corresponding Author

jake102518@yonsei.ac.kr, glee@yonsei.ac.kr

Abstract -

More than 500 statutes and rules are interlinked with building codes and regulations, with 24 amendments introduced in 2023. This complex network of references, combined with frequent revisions, poses significant challenges for both legal experts and laypeople in interpreting building codes. To address this, prior research has explored the application of retrieval-augmented generation (RAG) techniques to large language models (LLMs). However, these efforts have struggled to retain contextual information due to data segmentation. This study proposes and evaluates six variants of zero-shot Chain-of-Thought (CoT) prompts, leveraging long-context window (LCW) LLMs for retrieving relevant legal texts. Among these, the prompt requiring minimal human guidance achieved 63.16% accuracy on a dataset of 171 legal interpretative question-answer (LIQA) pairs, outperforming other variants by up to 5.7%p. Furthermore, incorporating a high-level community summary improved the model's robustness to prompt variations. This approach reduced the need for extensive document hops when locating relevant provisions, narrowing the performance gap to 1.17%p. Lastly, the LCW LLM consistently referenced definition provisions, ensuring greater consistency in legal interpretation. The less the guidance, the better the performance. Less is Better!

Keywords -

Legal interpretation; Building code; Long context window; Multi-hop reasoning

1 Introduction

Legal interpretation involves clarifying the meaning and intent of legal texts, such as building codes and regulations, to ensure their accurate and proper application to specific cases. Understanding these codes is challenging, even for professionals, due to their complexity and frequent updates. In South Korea, more than 500 statutes and rules are intricately linked to building codes and regulations [1], with 24 amendments introduced in 2023 [2]. This complexity often requires reviewing over 400 provisions to process a single legal action such as issuing building permits [3]. The scale of this challenge is evident in the over 900,000 legal interpretation petitions related to housing and construction filed annually in South Korea [4]. Notably, a study conducted in 2017 revealed that 96.7% of handwritten petitions focused on legal interpretation issues [5].

Recent studies have leveraged large language models (LLMs) to minimize the manual effort and cost involved in legal interpretation. To enhance the capabilities of vanilla LLMs, three techniques have been commonly investigated: fine-tuning, retrieval-augmented generation (RAG), and prompt engineering. Among the three methods, RAG-based approaches are widely employed to deliver up-to-date factual information and mitigate hallucinations [1, 6, 7]. However, their performance depends significantly on the structure of the underlying database, emphasizing the need for a database designed to optimize efficient information retrieval [1]. Despite these efforts, data must still be segmented into chunks, resulting in contextual loss. This limitation can lead to the retrieval of documents that contain relevant keywords but present them within unrelated or misleading contexts [7].

To mitigate RAG's context loss, long context window (LCW) LLMs have recently gained attention. By accommodating longer input sequences, LCW LLMs demonstrate advantages in capturing relationships between pieces of information spread across extensive text [8]. However, these models still face challenges, including limitations in context length and the "lost in the middle" problem, where they tend to prioritize information at the beginning and end of the input sequence [9].

To overcome the limitations of existing methods, we propose and compare six variants of zero-shot Chain-of-Thought (CoT) prompts to identify effective strategies for utilizing a long context window in complex tasks such as legal interpretation. These variants differ in their approaches to retrieving relevant information: unguided, exhaustive, and structured search. Exhaustive search allows the model to perform unlimited recursive searches within and across statutory documents, while structured search guides the model to retrieve information based on the hi-

erarchical structure of the legal texts. Additionally, we assess the impact of providing a high-level community summary (HLCS) [10], which includes the titles of each legal provision, for all prompt variants.

For validation, we constructed the *legal interpretative question-answer* (LIQA) dataset using the Building Statutes Question Answering (QA) document published by the Ministry of Government Legislation (MOLEG) [11]. This document contains 171 QA cases requiring interpretation of legal provisions, as answers are not explicitly stated within the legal texts. The LLM identifies relevant provisions using each prompt variant to generate answers, which are then evaluated for accuracy to assess prompt effectiveness.

This paper addresses various terms related to natural language processing (NLP). *Multi-hop search* refers to retrieving information needed to answer a user inquiry through logical steps applied to the provided documents [12]. Specifically, *intra-document hop* refers to hops occurring within a single document, while *cross-document hop* involves hops between multiple documents [13]. *Contextual reasoning* is a type of reasoning task that involves interpreting a small set of factual information specific to the given problem [14]. *High-level community summary* is a summary of a group of data to understand the global structure and semantics of the dataset [10]. For Korean building codes and regulations, we adhere to the English terms provided by the official English-Korean legal terminology system [15, 16]. Additionally, the term *provision* is used throughout this paper to collectively refer to both attached forms and articles, including their sub-hierarchies.

2 Relevant Study

2.1 Retrieval-Augmented Generation

To accurately address a legal inquiry, it is essential to first retrieve the relevant provisions from a large legal corpus. Zhong [17] noted that while legal experts tend to resolve legal tasks by building well-defined rule-based or symbolic systems, NLP researchers prioritize legal information retrieval to support data-driven interpretation. Recently, RAG has gained significant attention as a method to efficiently supply task-specific information to vanilla LLMs [6].

Savelka et al. [7] assessed GPT-4's ability to explain legal terms using RAG by providing court case data that illustrated how statutory terms had been previously interpreted. Their findings demonstrated that incorporating legal information retrieval improves the quality of LLM-generated interpretations, particularly in terms of factual accuracy, clarity, relevance, information richness, and precision. Similarly, Cho and Kim [1] developed the Semantic Processing for Architecture Regulation Compliance (SPARC) model, a RAG-based framework. Their evaluations showed that the system attained high accuracy rates for general inquiries (81.8%) and moderate accuracy for complex legal interpretation cases (56.1%).

2.2 Prompt Engineering and Long Context Window

Since language models have limitations in their input sequence length, prompt engineering is aimed at utilizing the available context window as efficiently as possible. Blair-Stanek et al. [18] tested GPT-3's ability to answer statutory reasoning questions about U.S. tax codes using zero-shot, few-shot, and CoT prompting. Their findings revealed that GPT-3 exhibited incorrect prior knowledge of the U.S. tax codes. However, two-shot prompting improved accuracy by an average of 20.9%p compared to zero-shot result. Fuchs et al. [19] evaluated GPT-3.5's ability to generate a structured XML format of building regulations under four conditions: few-shot, detailed description, CoT, and self-consistency prompting. The study demonstrated that LLMs can translate regulations into structured formats without fine-tuning.

Although prompt engineering optimizes the use of a given sequence length, ensuring a large context window is crucial for long documents [20]. Previous studies have focused on developing LLMs with an expanded context window [8] or proposing frameworks to increase the amount of information that can be processed within the existing limits [20, 21]. In February 2024, Google introduced Gemini 1.5 pro, an LCW LLM capable of handling up to 10 million tokens. According to the official report [8], the model achieved over 99% recall in multimodal information retrieval tasks.

2.3 Research Gaps and Questions

While RAG enhances information retrieval in LLMs, its application in legal interpretation has several limitations. First, retrieval methods such as similarity search often fail to capture essential information due to their inability to recognize contextual relationships between predivided data chunks. For example, a retrieved sentence might contain relevant legal terminology but appear in an irrelevant or misleading context [7]. Second, addressing these retrieval failures in complex tasks requires extensive pre-processing, including the creation of a well-structured database, which is both time-intensive and costly [1].

Moreover, despite their strong performance in information retrieval, LLMs with LCW also remain imperfect. One notable issue is the "lost-in-the-middle" problem, where the model disproportionately focuses on the beginning and end of a prompt, even when the context window has not been fully utilized [22]. Additionally, vanilla LCW LLMs often perform poorly on domain-specific tasks, such as legal interpretation.

To enhance the interpretation of building codes, which demand both comprehension and the ability to parse relevant provisions from extremely long contexts, this study evaluates various types of Chain-of-Thought (CoT) prompts using LCW LLMs. The following sections present the proposed prompt designs, the implementation methods, and the experimental evaluation process.

3 Proposed Method

We propose six variants of zero-shot prompts to enable the retrieval of provisions without the need for frameworks like RAG or fine-tuning. These retrieval prompts are grouped into three categories: unguided search, exhaustive search, and structured search. Each category includes two versions—one with a high-level community summary and one without—yielding a total of six variants.

3.1 General Prompt Structure

All prompts are created based on the structure outlined in *prompting guide 101* by Google Gemini Team [23]. The guide recommends incorporating four components in a prompt: persona, task, context, and format, though including all of them is optional. The persona defines the role or perspective, the task specifies the objective, the context provides situational details, and the format outlines the desired response style. The guide also highlights the importance of using a clear verb or directive to define the task. In this paper, we propose a basic prompt structure, as illustrated in Basic Prompt, where braces denote variable or optional elements specific to the three prompt variants.

Basic Prompt "You are designed to assist users in locating legal provisions related to the Korean Building Code as they apply to inquiries. {**HLCS**} {**Context**} {**Format**}"

3.2 Prompt Variants

3.2.1 Unguided Search

The prompt used in unguided search does not include detailed CoT instructions. Instead, it provides only a command in the {Context} field. Additionally, the response format is defined as tuples within a list, as described below.

{**Context**}: "From the given files and inquiry, find all articles and attached forms related to the inquiry."
{**Format**}: "Return the law name and article or attached form number that you found as tuples in a list format. For example: [(Building Act, Article 3-2), (Enforcement Rule of the Building Act, Article 12), (Enforcement Rule of the Building Act, Attached Form 1)]. If it is not an attached form, number must include the word 'Article'. Do not directly answer the inquiry."

3.2.2 Exhaustive search

In exhaustive search, LCW LLM is guided to perform a recursive provision search, hopping between all interreferencing provisions until it encounters provisions unrelated to the inquiry. Before returning the final list, it evaluates the relevance of each provision and includes only the necessary ones. To track the multi-hop search path, results are returned in a dictionary format, where the key represents the found provision and the value represents its parent provision.

{**Context**}: "Using the given files and inquiry: 1. Identify the articles and attached forms related to the inquiry. 2. Locate other provisions that reference the provisions identified in Step 1. 3. Recursively find additional provisions that reference those identified in the previous step until provisions unrelated to the inquiry are reached. In subordinate codes, references may appear explicitly, such as 'Article 12 Paragraph 1 of the Act'. 4. From the provisions gathered, filter and return only those necessary to address the inquiry."
{**Format**}: "5. Return the results in a dictionary format, where each key is a provision, and its value is the provision that mentioned the key (its parent). Each element must be formatted as (law name, article or form number). If the provision is related to the inquiry itself, its value should be 'Inquiry'. For example: '(Building Act, Article 3-2)': 'Inquiry', '(Enforcement Rule of the Building Act, Article 28)': '(Building Act, Article 3-2)', '(Enforcement Rule of the Building Act, Attached Form 1)': '(Building Act, Article 3-2)'. If the provision is not an attached form, the number must include the word 'Article'. Do not directly answer the inquiry."

3.2.3 Structured search

In structured search, the context part is organized according to the hierarchical structure of the Korean building codes. For each Act, specific provisions necessary for its enforcement are determined by the Decree, while provisions are only incorporated into the Rule if they pertain to technical or minor matters requiring frequent updates [24]. Consequently, the search process is structured to prioritize provisions in the order of Act, Decree, and Rule. However, since some inquiry topics are not explicitly addressed in the Act, the model is allowed to initiate searches from either the Act or the Rule. The response format is the same as that used in unguided search, as described in Section 3.2.1.

{**Context**}: "Using the given files and inquiry: 1. Identify the articles and attached forms in the Act or Rule related to the inquiry. 2. Locate other provisions in Rule that reference the provisions identified in Step 1. 3. Find additional provisions in Decree that reference those identified in the previous steps. In subordinate codes, references may appear explicitly, such as 'Article 12 Paragraph 1 of the Act'. 4. From the provisions gathered, filter and return only those necessary to address the inquiry."

3.3 High-level Community Summary

For each prompt variant, two versions—one with HLCS and one without—are tested. In this study, the titles of provisions are used as summaries, as they are designed to concisely convey what each provision regulates, facilitating convenient search [24]. The HLCS prompt is provided below.

{**HLCS**}: Each article's title is provided in a text file named after the law, followed by '(Provision Title)', which can help determine relevance.

4 Experimental Design

4.1 Datasets

The dataset used to validate our prompts is derived from the Building Statutes Question Answering (QA) document, which includes 171 statutory building code interpretation cases across 21 categories, selected by MOLEG [11]. This document is well-suited for use as ground truth in evaluating the prompts because MOLEG provides standardized legal interpretation guidelines in South Korea to ensure consistency across administrative agencies' interpretations [25]. Additionally, the cases in this document require interpreting the meaning and relationships between provisions, as the answers are not explicitly stated in the legal texts.

Each case in the document is structured into four parts: the title, an inquiry, a response, and the reasoning. As illustrated in Figure 1, we developed the LIQA dataset by using the inquiry summaries as questions and compiling the corresponding ground truth answers as follows: the title served as the inquiry summary, the response as the comprehensive conclusion, and the reasoning as the basis of judgment.

4.2 Experimental Setup

Figure 1 also illustrates the experimental flow for evaluating the six prompt variants. Each variant searches and generates a list of relevant provisions for each question in the LIQA dataset. The content of the provisions in each list is then passed to the answer-generating LLM to produce the final answer. The prompt for the answer-generating LLM is shown in Response Prompt.

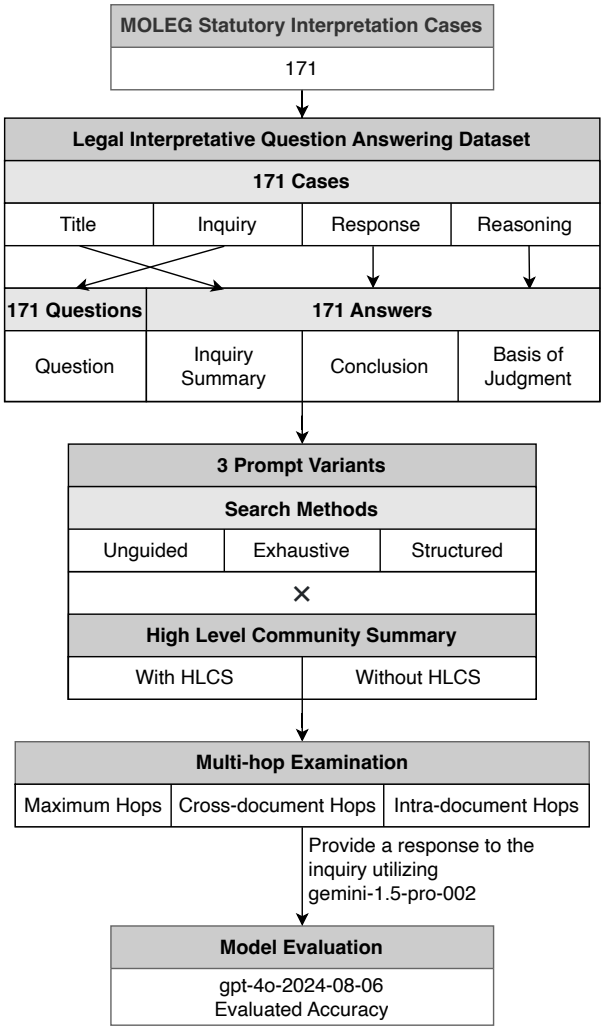

Figure 1. Flow chart of the experiment

Response Prompt You are designed to assist users in interpreting legal provisions related to the Korean Architecture Code. Your primary role is to provide interpretations of the laws as they apply to inquiries in Korean. You must analyze the legal texts in the provided documents or prompt only. The answer must consist of 3 parts: 'Inquiry Summary', 'Conclusion', 'Basis of Judgment'. The first part, 'Inquiry Summary', describes the core content of the inquiry. The second part, 'Conclusion', describes the answer or conclusion of the question. The last part, 'Basis of Judgment', contains the legal basis of the answer described in 'Conclusion'.

The proposed framework can accommodate any LCW LLM with a sufficiently large context window to handle the input data. For this experiment, we used the gemini-1.5-pro-002 model as the LCW LLM for CoT prompt processing and answer generation. To stay within context limits, provision search was restricted to the Building Act and its subordinate regulations. For hyperparameters, we followed Google AI Studio's defaults: *temperature* = 1, *top_p* = 0.95, *top_k* = 40, and *max_output_tokens* = 8192. Safety settings were disabled "threshold": "BLOCK_NONE") to prevent unintended response blocking. During answer generation, *temperature* was increased to 1.2 for enhanced creativity [26].

During answer generation, the content of the statutes is passed to the LLM as a string, retrieved via the Law Open Data Service through an Open API (hereafter referred to as LawAPI) provided by the Korean Law Information Center under MOLEG. For attached forms, however, the official PDF file of the form is provided to ensure the tabular format is preserved accurately.

The experiment was conducted on a MacBook Pro 14 (Model MacBookPro18,3) equipped with an Apple M1 Pro chip, 16 GB RAM, 10 CPU cores (8 performance cores and 2 efficiency cores), 12 GPU cores, and running macOS Sonoma 15.2. The application software used was Visual Studio Code version 1.96.2, with Python 3.11.5.

4.3 Accuracy Evaluation

The accuracy of each prompt variant was evaluated by comparing the final answer generated by the model to the ground truth answer. To ensure an efficient and fair evaluation, we employed a gpt-4o-2024-08-06 based assessment agent using the Evaluation Prompt. Q, R_H, and R_S represent a question, human response, and system response, braces indicating their variability by inquiry and prompt.

Evaluation Prompt Compare human_response and system_response for original_question. Return True if both responses reach the same conclusion or agree, otherwise return False.
original_question: {Q}
human_response: {R_H}
system_response: {R_S}

4.4 Multi-hop Examination

To trace the sequence of hops taken by the model during an exhaustive search, we instructed the LLM to return its results in a dictionary format, where the key represents the child (destination) and the value represents the parent (origin), as described in Section 3.2.2. The number of hops performed was then analyzed by converting the results into a tree format using Python code.

5 Results

5.1 Accuracy

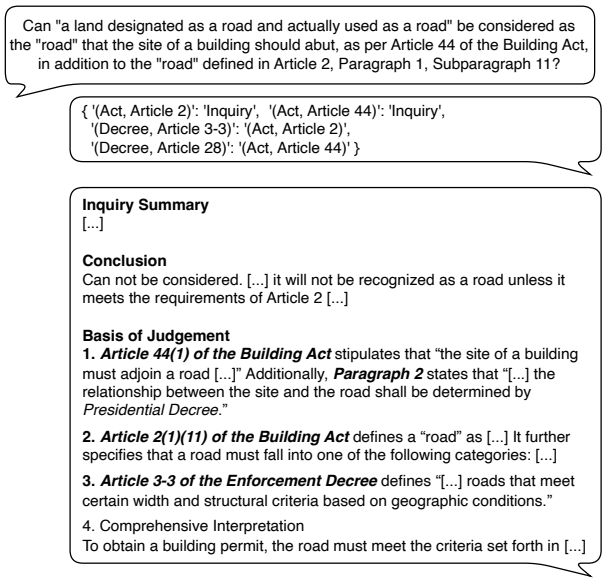

Figure 2. Example result of a Exhaustive Search + HLCS prompt.

Figure 2 depicts an example result when the exhaustive search prompt with HLCS is applied. Figure 3 presents the accuracy distribution for six prompt variants across different inquiry categories, with average accuracies represented by diamond marks and detailed in Table 1. Without HLCS, unguided search achieved the highest average accuracy of 63.16%, followed by exhaustive search at 60.82% and structured search at 57.89%. With HLCS, the accuracy gap between the prompt types narrowed, with structured search leading at 62.57%, followed closely by exhaustive (61.99%) and unguided (61.40%) search.

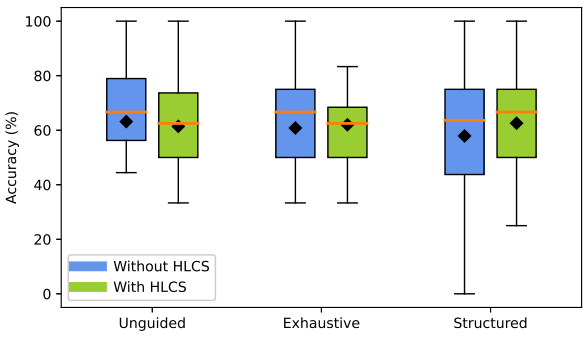

Figure 3. Comparison of six prompt variants on accuracy for each inquiry subject. Diamond marks indicate the weighted accuracy of each variant.

Table 1. Weighted average accuracy(%) of each prompt variant with and without HLCS. Ung, Exh, and Str stand for unguided, exhaustive, and structured search, respectively.

	Ung	Exh	Str
Without HLCS	**63.16**	60.82	57.89
With HLCS	61.40	61.99	62.57

5.2 Multi-hop Examination

Table 2 presents the top 5 most commonly mentioned provisions in the inquiries and the top 5 most frequently referenced provisions by the two exhaustive search prompts. Comparing the results reveals that Article 11 of the Building Act and Attached Form 1 of its Enforcement Rule are frequently both mentioned and referenced. Notably, the search prompts often referenced Article 2 of both the Act and the Rule, even though these were not explicitly mentioned in the inquiries. Article 2 in both cases contains definitions of the legal terms used in the content.

Table 2. List and count of the provisions commonly mentioned and searched. AF stands for Attached form. The number within parentheses represents frequencies (number of inquiries)

Idx	Mentioned	Exhaust / Exhaust+HLCS
1st	Act, 11 (25)	**Act, 2 (101)** / Rule, 2 (61)
2nd	Rule, AF 1 (20)	Rule, 2 (89) / Act, 2 (59)
3rd	Rule, 119 (15)	Rule, AF 1 (53 / 39)
4th	Rule, 86 (14)	Act, 11 (38) / Act, 11 (36)
5th	Act, 80 (10)	Decree, 6 (36) / Rule, 28 (30)

Figure 4 displays the percentile rank of the number of provisions referenced during exhaustive search, with and without HLCS. The plain exhaustive search referenced more provisions beyond the 97th percentile, with an average of 10.70 provisions compared to an average of 7.62 provisions for the HLCS-equipped search. Moreover, the HLCS-equipped LCW LLM performed fewer hops compared to the model without HLCS (Figure 5). The average length of the longest sequence of hops per inquiry was 2.91 for Exhaust+HLCS, which is 0.44 fewer than that of the model without HLCS (3.35). Specifically, it averaged 0.23 fewer cross-document hops (CDH) (1.62 vs 1.85) and 0.11 fewer intra-document hops (IDH) (1.22 vs 1.33) per inquiry.

Table 3 presents the average execution time of the six prompt variants for generating the list of relevant provisions. Exhaustive search took the longest time, both with and without HLCS, followed by structured search and unguided search. Meanwhile, the average prompt token count was similar across models—546K without HLCS and 556K with HLCS—since the majority of tokens came from the provided Building Act and its subordinate regulations.

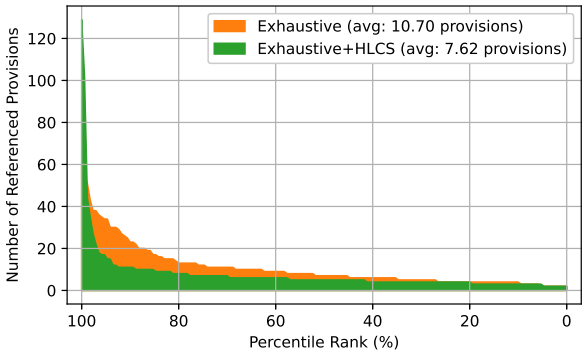

Figure 4. Comparison of the number of referenced provisions with and without HLCS across percentile ranks

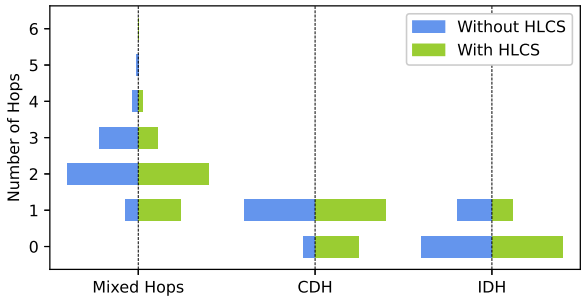

Figure 5. Horizontal bar plot illustrating the relative frequency of the number of hops taken by the exhaustive search.

Table 3. Average execution time(sec) of each prompt variant with and without HLCS.

	Ung	Exh	Str
Without HLCS	44.08	67.10	54.83
With HLCS	44.55	**83.06**	76.23

6 Discussion

When analyzing the trend in average response accuracy across six prompt variants, a counterintuitive pattern emerged: less guidance in the prompt for locating relevant provisions led to higher accuracy. Furthermore, incorporating HLCS improved the accuracy. Without HLCS, the accuracy gap between prompts reached up to 5.27 percentage points, whereas with HLCS, this gap narrowed to just 1.17 percentage points—a reduction of approximately 4.5 times. As detailed in Section 3.3, we provided provision titles as HLCS, and the results confirm their effectiveness

in legal document retrieval. However, while HLCS enhanced stability, it did not yield the highest performance, indicating that its primary effect is to ensure consistency in search results rather than directly enhancing accuracy.

HLCS's stability enhancement is further examined by the findings in Section 4.4. When the same prompt was used, the inclusion of HLCS reduced the average number of referenced legal provisions per inquiry by 3.08. Figure 5 shows HLCS reduced the length of multi-hop searches, making one- or two-hop searches predominant. Notably, the average number of cross-document hops decreased by 0.23, a more substantial reduction than the 0.11 decrease in intra-document hops. These results suggest that HLCS helps minimize unnecessary navigation and exploration across different documents. However, as shown in Table 3, using HLCS significantly increased execution time—rising by a factor of 1.24 in the case of exhaustive search—even though token usage was only 1.02 times higher. This may be attributed to the LLM repeatedly referencing the entire HLCS for each hop.

Table 2 showed that Article 2 of the Building Act and its Enforcement Decree, which are both definition provisions, were the most referenced. Exhaustive search prompt referred definition provisions in 59.1% of the inquiries. As noted by MOLEG [16], definition provisions are designed to clarify the meanings of terms within laws and regulations, ensuring their interpretation and application remain unambiguous. The frequent reference to these provisions suggests that the LCW LLM has the ability to provide consistent legal interpretations even when presented with new inquiries. Since we did not fine-tune or apply RAG with dataset-specific information, our findings may extend to other legal systems or domains.

7 Conclusion

This study designed and evaluated six prompt variants for LCW LLM-based legal interpretation, focusing on the unique challenges posed by the cross-referencing and hierarchical structure of legal texts. Notably, the exhaustive prompt aligned with the cross-referencing structure of laws, while the structured search prompt targeted hierarchical legal frameworks. Additionally, we investigated the influence of HLCS on model performance.

For evaluation, we developed the LIQA dataset, comprising 171 building code interpretation cases curated by MOLEG. Our findings revealed that providing less guidance to the model reduced the number of referenced provisions, improving response accuracy by up to 5.7%p. While HLCS helped stabilize model performance by narrowing the performance gap to 1.17%p, it did not maximize accuracy. Furthermore, the LCW LLM frequently referenced definition provisions, up to 59.1% of total inquiries, which contributed to consistency in legal interpretation.

Our research offers three key contributions. First, it demonstrates that minimizing human guidance enhances model performance when interpreting extensive and intricately cross-referenced legal data. Second, it highlights how the HLCS enhances stability across various prompt designs, reducing result variability and thereby lowering the manual effort and costs associated with LLM-based legal interpretation. Lastly, the LCW LLM consistently referenced definition provisions, contributing to more consistent and reliable interpretation results.

Future work could explore alternative LLM techniques, such as fine-tuning or RAG, to further optimize performance. Additionally, since the model typically requires fewer than three hops to locate relevant provisions, predefining the top three related provisions could help reduce token costs while maintaining high accuracy in legal interpretation tasks. Lastly, the designed guidance methods should be evaluated across various LLMs and multiple legal QA datasets to assess their generalizability, along with a qualitative comparison of the generated answers.

Acknowledgments

This work is supported in 2025 by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant RS-2021-KA163269 and RS-2024-00407028) and the Hans Fischer Senior Fellowship program at the Technical University of Munich - Institute for Advanced Studies (TUM-IAS) in Germany.

References

[1] S.-K. Cho and S.-S. Kim. *Study on the Efficient Response to Architectural Civil Complaints Using Large Language Models (LLM)*, volume 1. Architecture Urban Research Institute, Sejong, 2023.

[2] Y.-B. Lee. *Geonchug Beobje Donghyang 2023 [Building Legislation Trends 2023]*, volume 1. Architecture Urban Research Institute, Sejong, 2023.

[3] MOLIT. Korean building code, attached form 1. On-line: `https://www.law.go.kr/œ/œμœ/(2024-206,20240419)`, Accessed: 01/13/2025.

[4] Y.-G. Lee, J.-G. Lee, M.-J. Kim, and Y.-E. Hong. *Hannun-e Boneun Geonchugminwon Bigdata 2021 [Construction Civil Complaints Big Data 2021 at a glance]*, volume 1. Architecture Urban Research Institute, Sejong, 2022.

[5] S.-Y. Ryu, M.-J. Kim, Y.-J. Cho, and K.-H. Yu. A study on the analysis of civil appeals to building regulations. *Journal of the architectural insti-*

tute of Korea planning design, 33:13–21, 2017. doi:10.5659/JAIK_PD.2017.33.2.13.

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. doi:10.48550/arXiv.2005.11401.

[7] K.D. Gray M.A. Westermann H. Xu H. Savelka, J. Ashley. Explaining legal concepts with augmented large language models (gpt-4). In *arXiv*, Preprint. doi:10.48550/ARXIV.2306.09525.

[8] Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. In *arXiv*, Preprint. doi:10.48550/arXiv.2403.05530.

[9] K. Hewitt J. Paranjape A. Bevilacqua M. Petroni F. Liang P. Liu, N.F. Lin. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12: 157–173, 2024. doi:10.1162/tacl_a_00638.

[10] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization. In *arXiv*, Preprint. doi:10.48550/arXiv.2404.16130.

[11] Korea Institute of Registered Architects. *GeonChug-BeobLyeong Jil-UiHoeSinJib [Building provision inquiry collection]*, volume 1. MOLIT and KIRA, Seoul, 2023.

[12] W. Xiong, X. Li, S. Iyer, J. Du, P. Lewis, W.Y. Wang, Y. Mehdad, S. Yih, S. Riedel, D. Kiela, and B. Oguz. Answering complex open-domain questions with multi-hop dense retrieval. In *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*, pages 2590–2602, Hong Kong, China, 2024.

[13] K. Lu, I.-H. Hsu, W. Zhou, M.D. Ma, and M. Chen. Multi-hop evidence retrieval for cross-document relation extraction. In *Findings of the Association for Computational Linguistics (ACL)*, pages 10336–10351, Toronto, Canada, 2024.

[14] F. Giunchiglia. Contextual reasoning. *Epistemologia*, 16:345–364, 1993.

[15] KLRI. Glossary, english law system. On-line: `https://elaw.klri.re.kr/eng_service/lawsystem.do`, Accessed: 10/06/2024.

[16] MOLEG. Beoblyeong ib-ansimsa gijun [standards for drafting and reviewing laws]. On-line: `https://www.lawmaking.go.kr/lmKnlg/jdgStd/list`, Accessed: 10/06/2024.

[17] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun. How does nlp benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. doi:10.18653/v1/2020.acl-main.466.

[18] Blair-Stanek, A. Holzenberger, and B. N. Van Durme. Can gpt-3 perform statutory reasoning? In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, pages 22–31, New York, USA, 2023.

[19] S. Fuchs, M. Witbrock, J. Dimyadi, and R. Amor. Using large language models for the interpretation of building regulations. In *arXiv*, Preprint. doi:10.48550/arXiv.2407.21060.

[20] S. Chen, S. Wong, L. Chen, and Y. Tian. Extending context window of large language models via positional interpolation. In *arXiv*, Preprint. doi:10.48550/arXiv.2306.15595.

[21] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, and F. Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems (NeuralPS)*, 2023. doi:10.48550/arXiv.2306.07174.

[22] N.F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12: 157–173, 2024. doi:10.1162/tacl_a_00638.

[23] Gemini Team. *Prompting guide 101*, volume 2024 edition. Gemini for Google Workspace, California, USA, 2024.

[24] MOLEG. *Beoblyeong Ib-anSimsa Gijun [Standards for Drafting and Reviewing Laws]*. MOLEG, Sejong, South Korea, 2023.

[25] MOLEG. Beoblyeong haeseog annae [about legal interpretation], legislative work information (2024). On-line: `https://www.moleg.go.kr/menu.es?mid=a10106010000`, Accessed: 01/13/2025.

[26] Gemini Team. About generative models. On-line: `https://ai.google.dev/gemini-api/docs/models/generative-models`, Accessed: 02/27/2025.