Layer-of-Thoughts Prompting (LoT): Leveraging LLM-Based Retrieval with Constraint Hierarchies

Wachara Fungwacharakorn[*] , **Nguyen Ha Thanh**[*] , **May Myo Zin**[*] , **Ken Satoh**[*]

Center of Juris-Informatics, ROIS-DS, Tokyo, Japan

{wacharaf, maymyozin, nguyenhathanh, ksatoh}@nii.ac.jp

Abstract

This paper presents a novel approach termed Layer-of-Thoughts Prompting (LoT), which utilizes constraint hierarchies to filter and refine candidate responses to a given query. By integrating these constraints, our method enables a structured retrieval process that enhances explainability and automation. Existing methods have explored various prompting techniques but often present overly generalized frameworks without delving into the nuances of prompts in multi-turn interactions. Our work addresses this gap by focusing on the hierarchical relationships among prompts. We demonstrate that the efficacy of thought hierarchy plays a critical role in developing efficient and interpretable retrieval algorithms. Leveraging Large Language Models (LLMs), LoT significantly improves the accuracy and comprehensibility of information retrieval tasks.

1 Introduction

In recent years, there has been an explosion of interest in various prompting techniques for leveraging Large Language Models (LLMs). However, many existing works provide frameworks that are overly general, lacking the specificity needed to tackle detailed prompt-related challenges. One critical aspect that has been largely overlooked is the differentiation between prompts in multi-turn interactions.

Our paper seeks to fill this gap by examining the importance of hierarchical relationships among prompts. Specifically, we introduce the concept of thought hierarchies to filter and refine candidate responses, creating more structured and explainable retrieval processes. We believe that the strength of thought hierarchy is a pivotal factor in designing algorithms that are both efficient and interpretable. By incorporating constraint hierarchies within LoT, our approach not only enhances retrieval accuracy but also addresses the scalability and comprehension issues associated with complex queries.

The LoT framework extends the Graph-of-Thoughts by representing reasoning as a graph where nodes, called *thoughts*, denote reasoning steps. These thoughts are partitioned into layers, and categorized into **layer thoughts**, which handle conceptual step given by the user, and **option thoughts**, which assist in finding solutions.

---

[*]These authors contributed equally

The process starts with initializing layer thoughts, and proceeds from the first layer down to the last. Each layer thought receives input from the prior layer and branches into option thoughts, generating partial solutions. Aggregated outputs from option thoughts are passed to the next layer or used to expand thoughts dynamically until the last layer is reached.

For retrieval tasks, the LoT framework utilizes hierarchical levels to filter and rank documents from a given corpus based on a query. Relevance scores can be aggregated using several metrics, ensuring efficient and effective document ranking. Outputs at each layer progressively filter documents, providing clear explanations based on aggregated scores to justify document relevance.

2 Related Work

2.1 Traditional Information Retrieval

Before neural networks became widely popular, classical NLP approaches were the go-to methods for solving information retrieval tasks (Cooper 1971; Luhn 1957; Salton and Buckley 1988). These techniques primarily employed various lexical matching strategies. The researchers proposed both logical and statistical models to measure the similarity between queries and potential matches. Despite their advantages—such as fast computation and versatility—these approaches mainly relied on text morphology. Since morphological similarity does not necessarily equate to semantic similarity, achieving high accuracy in semantic similarity posed a challenge. As a result, the performance of these methods is limited, particularly in cases where document-query pairs exhibit non-overlapping text but semantic relevance, or overlapping text lacking semantic relation.

2.2 Neural Information Retrieval

Early pre-trained neural network models, including pre-trained word embeddings such as Word2Vec (Mikolov et al. 2013), GloVe (Pennington, Socher, and Manning 2014), and FastText (Mikolov et al. 2018), have proven highly effective in capturing semantic relationships between words. In the legal domain, a significant advancement is Law2Vec (Chalkidis and Kampas 2019), a specialized word embedding trained on legal texts, which has demonstrated substantial effectiveness.

More recently, Transformer-based models (Vaswani et al. 2017) have established new performance benchmarks across various tasks. Notable examples include BERT (Devlin et al. 2019), BART (Lewis et al. 2020), the GPT series (Radford et al. 2018; Radford et al. 2019), and Sentence-BERT (Reimers and Gurevych 2019), which have achieved state-of-the-art results in general applications. In legal text analysis, these advancements have been mirrored by models such as Legal-BERT (Yilmaz et al. 2019; Rabelo et al. 2020; Yoshioka, Aoki, and Suzuki 2021; Nguyen et al. 2022a). Nguyen et al. suggest using attentive neural network-based text representation for statute law document retrieval, showing effective results in this area (Nguyen et al. 2022b).

2.3 Reasoning Topologies for LLM

The field of LLM prompting has seen a surge in interest regarding reasoning topologies, also known as X-of-Thoughts (Besta et al. 2024b; Liu et al. 2023). A seminal work in this area is Chain-of-Thoughts (Wei et al. 2022), which instructs LLMs to not only generate an output but also reveal their step-by-step reasoning process (each step is called a *thought*). This concept was extended to Tree-of-Thoughts (Yao et al. 2024), where LLMs can branch into multiple potential thoughts after each step and evaluate their progress towards the solution. Subsequently, Graph-of-Thoughts (Besta et al. 2024a) generalized this further, allowing each thought to connect with others. Connections can involve aggregating outputs from other thoughts or refining the thought itself. Graph-of-Thoughts presents a model to formalize topological prompting using four components:

1. **Reasoning process**: The reasoning process is represented as a directed graph. Each node in the graph represents one reasoning step producing a piece of information towards a solution. A direct edge from one thought to another represent that an output (the piece of information) from the start node is considered as an input of the end node.

2. **Thought transformation**: The thought transformation is a transition function (probably involving LLMs) that gives next plausible reasoning processes from the current one. The function includes adding new thoughts and edges based on the existing thought (e.g., branching or aggregating) or adding new edges without adding new thoughts (e.g., refining).

3. **Evaluation function**: This function evaluates thoughts in the reasoning progress as a score towards the solution, and LLMs may involve in this function. Sometimes, an evaluation function needs to consider the whole reasoning process because a score of one thought may be relative to others.

4. **Ranking function**: This function ranks thoughts to consider a sole output for the main task. This function is related to the evaluation function as it mostly rank thoughts according to the scores.

2.4 Constraint Hierarchies

LoT Prompting in this paper is inspired by constraint hierarchies (Borning, Freeman-Benson, and Wilson 1992), which

are used for modelling constraints with strengths. In constraint hierarchies, each constraint is evaluated by an error function $e(c, \theta)$ that returns a non-negative real number indicating how nearly constraint c is satisfied for a valuation θ; and the function returns a zero value when the c is exactly satisfied. The strengths of the constraints are indicated by a non-negative integer. Conventionally, constraints with strength level 0 are hard constraints (or *required constraints*) and constraints with strength level $i \geq 1$ are soft constraints (or *preferential constraints*) where a higher value of the strength indicates that the constraint is weaker. Following this, we can represent a constraint hierarchy H as a partition $H_0, H_1, \ldots, H_\ell$ where each H_i, called a *level*, contains all of the constraints in H with strength i.

Solutions to constraint hierarchies must satisfy all hard constraints and satisfy soft constraints as much as possible. That is, there is no valuation outside the solutions that better satisfies the soft constraints and satisfies all of the hard constraints. To consider which valuation better satisfies the soft constraints, constraint hierarchies introduce comparators, which are later extended into level comparators and hierarchical comparators (Hosobe and Matsuoka 2003). Given valuations $\theta, \theta', \theta''$, a level H_i of a constraint hierarchy H with strength i, the level comparator $\preceq_i$ must satisfy the following conditions

1. If $e(c, \theta) = e(c, \theta'')$ for every $c \in H_i$, then $\theta \preceq_i \theta'$ if and only if $\theta'' \preceq_i \theta'$

2. If $e(c, \theta') = e(c, \theta'')$ for every $c \in H_i$, then $\theta \preceq_i \theta'$ if and only if $\theta \preceq_i \theta''$

3. If $e(c, \theta) \leq e(c, \theta')$ for every $c \in H_i$, then $\theta \preceq_i \theta'$

4. If $\theta \preceq_i \theta'$ and $\theta' \preceq_i \theta''$, then $\theta \preceq_i \theta''$

The first and second conditions state that if errors after applying valuations are the same for every constraint in the level, then the level comparator works the same. The third condition states that if errors after applying one valuation is less than or equal to the errors after applying another for every constraint in the level, then the former is better than or equal to the latter. The forth condition indicates the transitive property of the level comparator (we omit one condition from (Hosobe and Matsuoka 2003) as we consider only one constraint hierarchy each time). For a level comparator $\preceq_i$, we write

- $\theta \sim_i \theta'$ if $\theta \preceq_i \theta'$ and $\theta' \preceq_i \theta$;

- $\theta \prec_i \theta'$ if $\theta \preceq_i \theta'$ and $\theta' \npreceq_i \theta$; and

- $\theta \not\sim_i \theta'$ if $\theta \npreceq_i \theta'$ and $\theta' \npreceq_i \theta$.

If $\theta \not\sim_i \theta'$ is not possible (meaning that, we have either $\theta \sim_i \theta'$, $\theta \prec_i \theta'$, or $\theta' \prec_i \theta$), $\preceq_i$ is said to be *total*. If $\theta \preceq_i \theta'$ implies that $e(c, \theta) \leq e(c, \theta')$ for every $c \in H_i$, $\preceq_i$ is said to be *local*. A local $\preceq_i$ is generally not total because there might be a level H_i with two constraints c_1, c_2 such that $e(c_1, \theta) > e(c_1, \theta')$ and $e(c_2, \theta) < e(c_2, \theta')$ and hence $\theta \not\sim_i \theta'$. If there is a numerical aggregation function $g(\theta, H_i)$ such that $g(\theta, H_i) \leq g(\theta', H_i)$ if and only if $\theta \preceq_i \theta'$, then $\preceq_i$ is said to be *global*. A global $\preceq_i$ is always total because a numerical comparator is total.
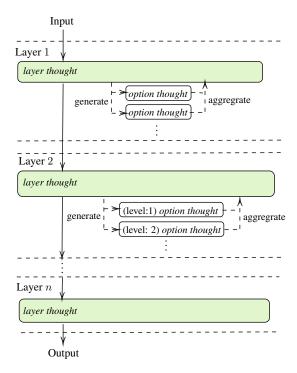
Figure 1: LoT Prompting

A hierarchical comparator $\prec$ covers overall comparisons by considering level by level, that is $\theta \prec \theta'$ if there exists $k \in \{1, \ldots, \ell\}$ such that $\theta \prec_k \theta'$ and for $i < k$, $\theta \sim_i \theta'$. Specific types of hierarchical comparators can be defined based on types of level comparators and aggregation functions used. Given the weight for constraint c is denoted by w_c, the original paper of constraint hierarchies (Borning, Freeman-Benson, and Wilson 1992) provides some examples of (hierarchical) comparators as follows.

- *locally-better* is a hierarchical comparator where a local-level comparator is used.

- *weighted-sum-better* is a hierarchical comparator with an aggregation function $g(\theta, H_i) = \Sigma_{c \in H_i} w_c e(c, \theta)$.

- *worst-case-better* is a hierarchical comparator with an aggregation function $g(\theta, H_i) = max(\{w_c e(c, \theta) | c \in H_i\})$.

- *least-squares-better* is a hierarchical comparator with an aggregation function $g(\theta, H_i) = \Sigma_{c \in H_i} w_c e(c, \theta)^2$.

3 LoT Prompting

Figure 1 illustrates Layer-of-Thoughts (LoT) prompting. The approach extends Graph-of-Thoughts (Besta et al. 2024a), which represents the reasoning process as a graph. The graph contains nodes, called *thoughts*, representing reasoning steps. Each thought receives outputs from the previous thoughts, and utilizes them to instruct LLMs in generating a partial solution for the next thoughts. In LoT Prompting, each thought is assigned a number to show which *layer* they are in. There are two types of thoughts in the graph.

- **Layer thought**: Each layer contains a layer thought to handle conceptual step given by the user. A layer thought in layer i has a previous layer thought in layer $i - 1$ and a next layer thought in layer $i + 1$, except the first and the last layer thoughts. A layer thought has a role to receive outputs from the previous layer thought, determine generating option thoughts, aggregate outputs from the generated option thoughts, and forward the aggregated output to the next layer thought.

- **Option thought**: Each layer contains multiple option thoughts or even none of them (e.g., the last layer in Figure 1). An option thought in layer i receives the inputs from the layer thought in the same layer, and generates a partial solution as the output for aggregating in the layer thought. A layer thought may generate equal option thoughts (e.g., Layer 1 in Figure 1) or prioritize option thoughts into several *levels*, defined by a positive integer (e.g., Layer 2 in Figure 1). We consider a layer with equal option thoughts as *single-level* layer and a layer with prioritized option thoughts as *multiple-level* layer

Thoughts in LoT Prompting are initiated as follows.

1. A user designs conceptual steps for the task.

2. Starting from the first step, each step initiates a new layer thought. The layer thought receives inputs from the previous layer thought (or from the main task for the first layer thought), and instructs LLMs to suggest appropriate criteria for that step. The criteria can be generated equally or prioritized in order of importance.

3. For each criterion, an option thought is generated. Each option thought instructs LLMs or performs calculation, and then provides a partial solution for the task according to the criterion.

4. The layer thought then aggregates the outputs from option thoughts in the layer, and determines whether the aggregated output is appropriate. If it is appropriate, then the output is fed to the new layer thought (invoke Step 2 for initiating a new layer thought from the next step). If it is not appropriate (e.g., no inputs from the previous layer pass the criteria), we can instruct LLMs to refine the criteria and initiate new option thoughts again (backtrack to Step 3).

3.1 Retrieval Tasks

In this paper, we explore leveraging LoT Prompting for retrieval tasks. Given a query q and a document corpus D, retrieval tasks aim to find the most relevant documents in D with respect to the query q. Retrieval tasks are usually evaluated using relevance scores. In LoT Prompting, we assume that each option thought can be evaluated using a non-negative relevance score $\rho(d, t)$ of a document $d \in D$ for an option thought t (greater score indicates more relevance). A relevance function is *binary* if the score can only be either 0 or 1. That is, the function returns 1 if the document d passes the criterion corresponding to the thought t and returns 0 if it fails.

We suggest several metrics to aggregate outputs from option thoughts in a single-level layer as follows.

- `all`: In this metric, relevance scores of documents in the aggregated output must be greater than 0 for every option thought within this layer. This is inspired from hard constraints in constraint hierarchies.

- `at-least-k` (where k is a positive integer): In this metric, relevance scores of documents in the aggregated output must be greater than 0 for at least k option thoughts within this layer. This metric is relaxed from `all`.

- `locally-better`: This metric is a relative metric, inspired by *locally-better* in constraint hierarchies. Some studies refer to this metric as Pareto efficiency. In this metric, the aggregated output contains only the top-ranked documents from the input, where no input document surpasses the relevance score of the top-ranked document for every option thought within this layer.

- `max-count`: Since `locally-better` is not a total order, `max-count-better` is a total order relaxed from that metric. `max-count-better` can only be used for binary relevance functions. In this metric, the aggregated output contains documents that maximize the number of passed criteria.

- `max-weight`: This metric is inspired by *weight-sum-better* in constraint hierarchies. Given the weight for the thought t as w_t, the aggregated output contain documents that maximize $\Sigma_{t \in T^i} w_t \rho(d, t)$ where T^i be the set of all of option thoughts in the layer i.

For `max-count` and `max-weight`, they do not need to restrict the aggregated output for only documents that maximize such metrics, but the layer thought can present scores of each input document for ranking instead. We call them `rank-count` and `rank-weight` respectively. These metrics can be extended for a multiple-level layer by aggregating outputs from the strongest level first and weaker levels successively.

One goal of LoT prompting is to reduce computations by skipping the need to explore the entire corpus with all option thoughts. In LoT prompting, option thoughts are partitioned by layers with layer thoughts that determine if further exploration is necessary. Let T denote the set of all option thoughts, n denote the total number of layers, m_i denote the number of levels in layer i (m_i can vary across layers), and T_i denote the set of all option thoughts in layer i. A *hierarchy* of T^i refers to a tuple $\langle T_1^i, \dots, T_{m_i}^i \rangle$, where T_j^i contains all of the option thoughts in the layer i with level j. T can be represented as a nested hierarchy:

$$\langle \langle T_1^1, \dots, T_{m_1}^1 \rangle, \langle T_1^2, \dots, T_{m_2}^2 \rangle, \dots, \langle T_1^n, \dots, T_{m_n}^n \rangle \rangle$$

It can be observed that the multiple-layer prompting has the same expessive power as a single layer with multiple levels because the nested hierarchy works in the same way as the flatten hierarchy:

$$\langle T_1^1, \dots, T_{m_1}^1, T_1^2, \dots, T_{m_2}^2, \dots, T_1^n, \dots, T_{m_n}^n \rangle$$

This indicates a structured arrangement of thoughts, with layers representing distinct blocks of thoughts to convey semantic understanding, and levels representing more specific evaluation criteria. The observation helps simplifying the explanation of the reasoning process. For example, assuming we explain an output with a set $E = \{t \in$

$T \mid \rho(d, t) > 0\}$. Consequently, E can be represented as $\langle E_1^1, \dots, E_{m_1}^1, E_1^2, \dots, E_{m_2}^2, \dots, E_1^n, \dots, E_{m_n}^n \rangle$, where E_j^i contains all of the option thoughts in E in the layer i with level j.

4 Application-Driven Experimental Setup

In this section, we present two experimental applications corresponding to two specific use cases: Japanese Civil Law retrieval and normative sentence retrieval. Through these applications, we verify the efficacy of the Layer-of-thought approach and identify its limitations.

4.1 Japanese Civil Law Retrieval

Legal AI is a niche field that leverages computer and artificial intelligence techniques to make legal processes more efficient and effective. The rapid progression of AI-based tools is set to emancipate legal professionals from cumbersome tasks such as document searches and contract reviews. The COLIEE competition (Goebel et al. 2024) is held annually to promote research in legal information processing. This competition includes various challenges like document retrieval and legal entailment.

Participants are provided with a legal question Q from the Japanese Bar Exam and are tasked with retrieving relevant articles $A1, A_2, \dots, A_n$ from the Japanese Civil Code. The ability to retrieve pertinent legal articles is crucial for several reasons. It enables legal professionals to quickly access relevant legal precedents and statutes, thereby enhancing their decision-making process. However, the task is particularly challenging due to the complex language and vast amount of information contained within legal texts.

The Japanese Civil Code is rich with intricate, specialized terminology, making the retrieval task demanding. Moreover, the volume of information makes it difficult to pinpoint the exact articles that are relevant to a given legal question. These challenges highlight the importance and difficulty of the task, making it a suitable candidate to test the feasibility of the Layer-of-Thoughts approach.

The evaluation metrics for this task include the macro average of F2, precision, and recall. This comprehensive set of metrics ensures a robust assessment of each team's performance. The calculations for these measures are as follows:

$$\text{Precision} = \text{avg} \left(\frac{\text{\# correct articles}}{\text{\# retrieved articles}} \right)$$

$$\text{Recall} = \text{avg} \left(\frac{\text{\# correct articles}}{\text{\# relevant articles}} \right)$$

$$\text{F2} = \text{avg} \left(\frac{5 \times \text{Precision} \times \text{Recall}}{4 \times \text{Precision} + \text{Recall}} \right)$$

The proposed design contains multiple layers following the idea of LoT (as demonstrated in Figure 2):

1. **Keyword Filtering Layer (KFL):** The first layer involves keyword filtering. The keywords are suggested by a Large Language Model (LLM) after it analyzes the query. This layer uses `at-least-1` metric.

Input: *(Example query: John, a 15 year old kid, made a contract ...)*
        + (A set of candidate legal articles)

- -
                                    Keyword Filtering Layer

From the query, please list keywords the article should have

Does the article contain a word $< minor >$?
Does an article contain a word $< contract >$?
            ⋮ (single - level)

- -

- -
                                    Semantic Filtering Layer

From the query, please list ordered criteria for the relevant articles

(level:1) Is the article relevant to $< contract\ law >$?
(level:2) Is the article relevant to $< juridical\ act >$?
            ⋮ (multiple - level)

- -

- -
                                    Final Confirmation Layer

Does the article indeed relevant to the query ?

- -
Output: Article 5 (1) A minor must obtain the consent of the minor's ...
         explanation: $\langle keyword : "minor", relevant : "juridical\ act" \rangle$
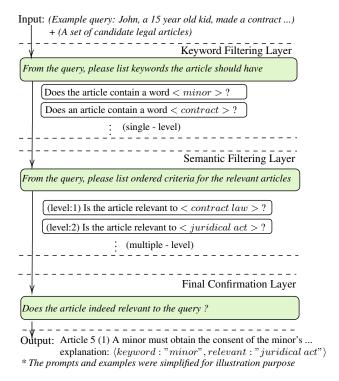** The prompts and examples were simplified for illustration purpose*

Figure 2: LoT Prompting in Japanese Civil Law Retrieval Setup

2. **Semantic Filtering Layer (SFL):** Following the keyword filtering, the semantic filtering layer applies conditions suggested by the LLM to the selected candidates. This is a multiple-level layer, organized from general to specific conditions, helping refine the selection progressively. This layer uses `max-count` metric.

3. **Final Confirmation Layer (FCL):** In the final layer, the original query is used once again. The LLM is asked to confirm whether the candidate articles can indeed help answer the query. This step ensures that only the most relevant articles are selected. This layer uses `all` metric.

Besides, we also challenge the proposed design with two simpler settings. In the first setting, the **SFL** is omitted, meaning that candidates are directly evaluated against the original query immediately after keyword filtering. In the second, even more minimal setting, we remove the **SFL** and **KFL** entirely.

Upon examining the results in Table 1, several insights can be drawn. The **Proposed Design following LoT** demonstrated the highest F2 score (0.835), outperforming the best systems in COLIEE 2024. This high F2 score indicates a well-balanced performance in terms of precision (0.838) and recall (0.839), showcasing the effectiveness of the Layer-of-Thoughts (LoT) approach. The structured filtering layers, including keyword and semantic filtering, ensure a balanced retrieval of relevant articles, minimizing false positives while maximizing true positives.

In contrast, the two direct validation systems exhibited higher recall but significantly lower precision. The **Direct Validation with Query (w. KFL)** system achieved a re-

call of 0.853 but only a precision of 0.546, resulting in an F2 score of 0.563. The **Direct Validation without Query (w/o. KFL)** system had a recall of 0.885 but a precision of merely 0.432, leading to an F2 score of 0.449. These results suggest a more tolerant approach, where the absence of semantic filtering layer causes an influx of irrelevant articles, thus lowering precision. An extreme instance highlighted this issue, where the LLM marked up to 400 articles as relevant while the gold standard identified only 2. This indicates a high risk of false positives in these settings. Combining filtering by keywords can be a simple yet effective strategy to reduce false positives.

Among the comparative models, **JNLP** achieved the highest F2 score (0.807) with good precision (0.709) and recall (0.870), followed by **CAPTAIN** with an F2 score of 0.800 (precision 0.732, recall 0.845). **TQM** also performed well with an F2 score of 0.782 (precision 0.785, recall 0.800). While these models showed strong overall performance, they did not surpass the balance achieved by the proposed design.

Other models such as **NOWJ**, **AMHR**, and **UA** displayed competent results with F2 scores of 0.772, 0.749, and 0.711, respectively. These scores indicate an effective yet slightly imbalanced performance compared to the top-performing models and the proposed design.

The high F2 score of the proposed design illustrates its superior balance between precision and recall, unlike the direct validation systems, which prioritize recall at the expense of precision. This balance is crucial in practical scenarios to ensure relevant legal articles are retrieved accurately, minimizing the effort required for further verification.

4.2 Normative Sentence Retrieval

In the context of automated driving, it is essential to have a comprehensive understanding of both explicit and implicit traffic rules. While explicit rules are often codified in statutes and regulations, implicit rules are frequently derived from judicial decisions and case law. These implicit rules, which are crucial for ensuring safe and lawful driving behavior, need to be systematically identified and made explicit.

Extracting normative sentences from court decisions is vital for developing a comprehensive corpus of traffic rules for automated driving systems. Advanced natural language processing (NLP) techniques, machine learning models, and expert validation can be employed to identify and articulate implicit rules, thereby enhancing the traffic regulation framework. This ensures that automated driving systems operate under a robust and legally sound set of traffic norms, promoting safety and compliance.

Currently, there are no annotated datasets available to train a model for retrieving normative sentences specifically related to Section 6 of the German Road Traffic Regulations (i.e., § 6 StVO)[1]. Normative sentences are those identified as additions, clarifications, or interpretations of the traffic rule stipulated by § 6 StVO. Developing rules or patterns to

---

[1]https://dejure.org/gesetze/StVO/6.html

System	Precision	Recall	F2
Proposed Design following LoT	0.838	0.839	0.835
Direct Validation with Query (w. KFL)	0.546	0.853	0.563
Direct Validation with Query (w/o. KFL)	0.432	0.885	0.449
Other Comparative Models			
JNLP (Nguyen et al. 2024a)	0.709	0.870	0.807
CAPTAIN (Nguyen et al. 2024b)	0.732	0.845	0.800
TQM (Li et al. 2024)	0.785	0.800	0.782
NOWJ (Nguyen et al. 2024c)	0.690	0.835	0.772
AMHR (Nighojkar et al. 2024)	0.651	0.825	0.749
UA (Housam et al. 2024)	0.610	0.800	0.711

Table 1: Comparison of Different Systems in Terms of Precision, Recall, and F2

accurately identify and differentiate these sentences is challenging because not every normative sentence is our focus; the sentence must be both normative and related to § 6 StVO. This challenge arises to distinguish between the broader category of normative sentences and those specifically pertinent to § 6 StVO. The complexity lies in the dual criteria of normativeness and relevance, which necessitates a more sophisticated approach than mere keyword matching or rule-based retrieval systems.

The evaluation dataset used in this study consists of English translations of 15 court decisions pertinent to § 6 StVO. We randomly selected these decisions from the *Case law on Section 6 StVO*[2] and translated them into English. The focus is on the reasoning sections of these decisions, as they are most pertinent for identifying normative statements. The dataset excludes other sections that do not contribute to understanding the normative content. Figure 3 shows a graphical representation of the total sentences and the normative sentences for each file ID.
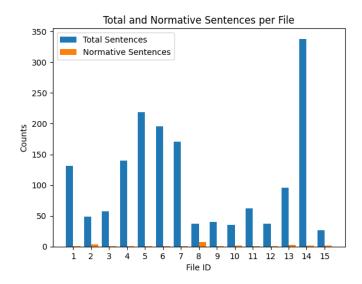

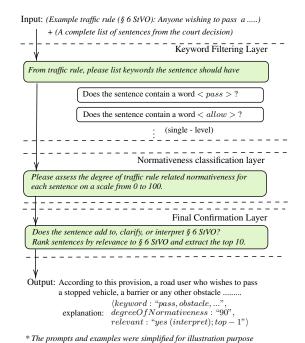Figure 3: Counts of Total and Annotated Normative Sentences per File

Figure 4: LoT Prompting in Normative Sentence Retrieval Setup

Minimizing manual effort in identifying normative sentences is a key objective. For each input file, if the system extracts 10 sentences and includes all normative sentences among them, we consider this an optimal outcome because it balances efficiency and comprehensiveness. Extracting a manageable number of sentences while ensuring all normative ones are included allows for quick verification and reduces the burden on manual review. Therefore, our evaluation metrics prioritize achieving higher recall to ensure that most, if not all, normative sentences are captured.

A multi-layered approach using LoT Prompting is proposed for extracting normative sentences, as demonstrated in Figure 4.

1. **Keyword Filtering Layer (KFL):** The first layer involves using a Large Language Model (LLM) to generate a list of relevant keywords associated with traffic rules. To ensure comprehensive coverage, word embed-

System	Precision	Recall	F2
Proposed Design following LoT	0.187	0.966	0.527
Other Comparative Models			
Chain-of-Thoughts	0.167	0.862	0.470
BM25 Retriever	0.153	0.793	0.432
Retrieve & Re-Rank	0.140	0.724	0.395

Table 2: Comparison of Normative Sentence Retrieval Systems in Terms of Precision, Recall, and F2

ding technique, such as Word2Vec, is applied to identify and match keywords or their closely related terms within sentences. This layer uses `at-least-2` metric, that is sentences containing at least two keywords that are either exact matches or semantically similar are selected for further processing.

2. **Normativeness Classification Layer (NCL):** Next, the second layer prompts the LLM to evaluate the filtered sentences. Each sentence is assessed based on its degree of normativeness, with normative sentences defined as those having a normativeness score of 70 or higher.

3. **Final Confirmation Layer (FCL):** The final layer involves the LLM verifying the relevance of the normative sentences to § 6 StVO. Each candidate sentence is cross-referenced with the specific description of § 6 StVO to ensure its relevance. This layer uses `rank-weight` metric to refine the selection process by narrowing down the results to the top 10 most relevant sentences, thereby focusing on the most pertinent information.

In this study, we employed GPT-4o as the LLM with parameters configured as follows: `temperature` at 0, `top-p` at 1, `frequency penalty` at 0, and `presence penalty` at 0.

Table 2 presents the results. Comparisons are made with three baseline methods: BM25 (Robertson and Walker 1994), Retrieve & Re-Rank[3], and the Chain-Of-Thoughts approach. BM25 is a widely used ranking function in information retrieval that measures the relevance of documents (or sentence, in this experiment) to a query based on term frequency and document length. Retrieve & Re-Rank involves an initial retrieval phase to identify a set of candidate sentences and a subsequent re-ranking phase to improve their relevance. In the experiment, the English translation of § 6 StVO serves as the query. The Chain-Of-Thoughts approach involves reasoning through sequences of thought or processing steps to extract information. Comparing with these methods helps evaluate the proposed approach's performance against established retrieval techniques and advanced reasoning models. Given that the number of normative sentences per file ranges from 1 to 7 (as indicated by Figure 3), focusing on extracting the top 10 sentences is justified. The top-k value is set to 10 for all models. The LoT method demonstrates precise and reliable extraction of normative sentences, achieving a higher recall score compared to the baselines. Although precision and F2 scores are not

exceptionally high, mainly because most files contain only one gold standard normative sentence while our method extracts ten sentences, the LoT method still outperforms the baseline approaches.

5 Discussion

The experimental results highlight the merits and potential drawbacks of the Layer-of-Thoughts (LoT) Prompting in various information retrieval tasks. For Japanese Civil Law retrieval, the LoT approach consistently outperformed traditional methods and even advanced retrieval models used in the COLIEE competition. This demonstrates that the integration of hierarchical thoughts and structured reasoning can significantly enhance the quality of retrieved documents by balancing both precision and recall. The high F2 scores indicate that LoT Prompting effectively captures the nuances of the task, leading to a more precise and well-rounded retrieval process.

In the context of normative sentence retrieval, the LoT method also showed promising results. Despite the inherently challenging nature of this task, which involves high variability and implicit rules, LoT managed to achieve a recall score close to 1.0. This high recall is particularly valuable in legal contexts, where missing relevant sentences could lead to critical oversights. However, the precision scores indicate that while most relevant sentences are captured, further refinement is needed to reduce false positives and enhance the overall relevance of the extracted sentences.

The scalability and efficiency of the LoT Prompting are largely supported by its hierarchical structure. The use of strength metrics to prioritize thoughts ensures that the computational overhead is managed effectively. By leveraging constraint hierarchies, LoT Prompting efficiently narrows down the search space, making it well-suited for large-scale information retrieval tasks. This structured approach not only reduces unnecessary computations but also improves the speed and accuracy of retrieving relevant documents.

One of the standout features of the LoT Prompting is its explainability. By breaking down the retrieval process into hierarchical levels of thoughts, LoT Prompting provides clear and interpretable pathways from the query to the final retrieved documents. Each step in the hierarchy can be traced and understood, offering transparency in how decisions are made. This interpretability is invaluable, particularly in legal and regulatory contexts, where the rationale behind document retrieval must be clear and defensible.

---

[3]https://sbert.net/examples/applications/retrieve_rerank/README.html

6 Conclusion

In this paper, we introduced the Layer-of-Thoughts (LoT) Prompting, which incorporates constraint hierarchies to enhance the retrieval of information through structured prompting. Our experiments on Japanese Civil Law retrieval and normative sentence extraction tasks demonstrated that the LoT Prompting significantly improves both precision and recall compared to baseline and state-of-the-art methods. The ability of LoT to balance these metrics while providing explainable and efficient retrieval processes makes it a compelling tool for complex information retrieval tasks. Future work will focus on further refining LoT Prompting to enhance precision, particularly in tasks involving highly variable and implicit information.

References

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Podstawski, M.; Niewiadomski, H.; Nyczyk, P.; and Hoefler, T. 2024a. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence* 38(16):17682–17690.

Besta, M.; Memedi, F.; Zhang, Z.; Gerstenberger, R.; Piao, G.; Blach, N.; Nyczyk, P.; Copik, M.; Kwasniewski, G.; Müller, J.; et al. 2024b. Demystifying chains, trees, and graphs of thoughts. *arXiv preprint arXiv:2401.14295*.

Borning, A.; Freeman-Benson, B.; and Wilson, M. 1992. Constraint hierarchies. *LISP and symbolic computation* 5(3):223–270.

Chalkidis, I., and Kampas, D. 2019. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. *Artificial Intelligence and Law* 27(2):171–198.

Cooper, W. S. 1971. A definition of relevance for information retrieval. *Information storage and retrieval* 7(1):19–37.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Goebel, R.; Kano, Y.; Kim, M.-Y.; Rabelo, J.; Satoh, K.; and Yoshioka, M. 2024. Overview of benchmark datasets and methods for the legal information extraction/entailment competition (coliee) 2024. In *JSAI International Symposium on Artificial Intelligence*, 109–124. Springer.

Hosobe, H., and Matsuoka, S. 2003. A foundation of solution methods for constraint hierarchies. *Constraints* 8(1):41–59.

Housam, K. B. B.; Md Abed, R.; Mi-Young, K.; Juliano, R.; and Randy, G. 2024. Legal yes/no question answering through text embedding, fine-tuning, and prompt engineering. In *Proceedings of the Eighteenth International Workshop on Juris-Informatics (JURISIN 2024)*, 170–180.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7871–7880. Online: Association for Computational Linguistics.

Li, H.; Chen, Y.; Ge, Z.; Ai, Q.; Liu, Y.; Zhou, Q.; and Huo, S. 2024. Towards an in-depth comprehension of case relevance for better legal retrieval. In *JSAI International Symposium on Artificial Intelligence*, 212–227. Springer.

Liu, T.; Guo, Q.; Yang, Y.; Hu, X.; Zhang, Y.; Qiu, X.; and Zhang, Z. 2023. Plan, Verify and Switch: Integrated Reasoning with Diverse X-of-Thoughts. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2807–2822. Singapore: Association for Computational Linguistics.

Luhn, H. P. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development* 1(4):309–317.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Mikolov, T.; Grave, E.; Bojanowski, P.; Puhrsch, C.; and Joulin, A. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Nguyen, H.-T.; Nguyen, M.-P.; Vuong, T.-H.-Y.; Bui, M.-Q.; Nguyen, M.-C.; Dang, T.-B.; Tran, V.; Nguyen, L.-M.; and Satoh, K. 2022a. Transformer-based approaches for legal text processing: Jnlp team-coliee 2021. *The Review of Socionetwork Strategies* 16(1):135–155.

Nguyen, H.-T.; Phi, M.-K.; Ngo, X.-B.; Tran, V.; Nguyen, L.-M.; and Tu, M.-P. 2022b. Attentive deep neural networks for legal document retrieval. *Artificial Intelligence and Law* 1–30.

Nguyen, C.; Tran, T.; Le, K.; Nguyen, H.; Do, T.; Pham, T.; Luu, S. T.; Vo, T.; and Nguyen, L.-M. 2024a. Pushing the boundaries of legal information processing with integration of large language models. In *JSAI International Symposium on Artificial Intelligence*, 167–182. Springer.

Nguyen, P.; Nguyen, C.; Nguyen, H.; Nguyen, M.; Trieu, A.; Nguyen, D.; and Nguyen, L.-M. 2024b. Captain at coliee 2024: large language model for legal text retrieval and

entailment. In *JSAI International Symposium on Artificial Intelligence*, 125–139. Springer.

Nguyen, T.-M.; Nguyen, H.-L.; Nguyen, D.-Q.; Nguyen, H.-T.; Vuong, T.-H.-Y.; and Nguyen, H.-T. 2024c. Nowj@ coliee 2024: leveraging advanced deep learning techniques for efficient and effective legal information processing. In *JSAI International Symposium on Artificial Intelligence*, 183–199. Springer.

Nighojkar, A.; Jiang, K.; Fields, L.; Bilgin, O.; Steinle, S.; Sadybekov, Y.; Marji, Z.; and Licato, J. 2024. Amhr coliee 2024 entry: legal entailment and retrieval. In *JSAI International Symposium on Artificial Intelligence*, 200–211. Springer.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Rabelo, J.; Kim, M.-Y.; Goebel, R.; Yoshioka, M.; Kano, Y.; and Satoh, K. 2020. Coliee 2020: Methods for legal document retrieval and entailment. In *New Frontiers in Artificial Intelligence: JSAI-IsAI 2020 Workshops, JURISIN, LENLS 2020 Workshops, Virtual Event, November 15–17, 2020, Revised Selected Papers*, 196–210. Berlin, Heidelberg: Springer-Verlag.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *The University of British Columbia Repository*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1(8):9.

Reimers, N., and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Robertson, S. E., and Walker, S. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*, 232–241. Springer.

Salton, G., and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24(5):513–523.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting: elicits reasoning in large language models. *Advances in neural information processing systems* 35:24824–24837.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems* 36.

Yilmaz, Z. A.; Wang, S.; Yang, W.; Zhang, H.; and Lin, J. 2019. Applying BERT to document retrieval with birch. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, 19–24.

Yoshioka, M.; Aoki, Y.; and Suzuki, Y. 2021. Bert-based ensemble methods with data augmentation for legal textual entailment in coliee statute law task. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, 278–284.