International Journal of Innovative Research in Computer Science and Technology (IJIRCST)
ISSN (Online): 2347-5552, Volume-14, Issue-6, Nov-Dec 2026
DOI: https://doi.org/10.55524/ijircst.2026.14.6.1
Article ID IRP-1795, Pages 1-13
www.ijircst.org

Developing a Taiwanese Legal Expert Response System Using Generative Artificial Intelligence and Retrieval-Augmented Generation

Yeu-Pong Lai[1], Chia-Ling Chen[2], and *Chen-Yu Li[3]

[1] Professor, Department of Information Management, Central Police University, Taoyuan, Taiwan
[2]Assistant Professor, Department of Maritime Police, Central Police University, Taoyuan, Taiwan
[3]Assistant Professor, Department of Information Management, Central Police University, Taoyuan City, Taiwan

Correspondence should be addressed to Chen-Yu Li; cyli@mail.cpu.edu.tw

Received: 21 Sep 2026; **Revised:** 6 Oct 2026; **Accepted:** 19 Oct 2026; **Published Online:** 1 Nov 2026

Copyright © 2026 *Chen-Yu Li et al. This is an open-access article distributed under the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

ABSTRACT- Recently, the Large Language Models (LLMs) have emerged as "knowledge consultants" in many fields, such as Natural Language Processing and Generation, Customer Service and Chatbots, Legal Field, Healthcare, Education Assistance, Financial Services, Content Creation, and Recommendation Systems and Search Engines. In the Legal Field, the LLMs can be applied in legal document analysis, case retrieve-al, contract review, and legal consultation to help lawyers and judges improve work efficiency and accuracy. This study aims to develop a legal expert question-answering system tailored for Taiwan's judicial sector by leveraging generative artificial intelligence and retrieval-augmented generation (RAG) techniques. By integrating a local LLM service with a vector database of judicial documents in certain domains, the system seeks to enhance the accuracy and efficiency of legal case investigation and consultation. With the advancement of Transformer architectures and LLMs such as BERT, LLMs have demonstrated outstanding language understanding and generation capabilities. However, they still face challenges like hallucinations and outdated knowledge. This research employs RAG technology to combine retrieval and generation, effectively reducing hallucinations and supplementing the latest information. The study uses 30 Chinese judicial rulings and indictments as data, applying Google Gemini and Meta LLaMA language models for text segmentation and vectorization. Experiments were conducted with various chunk sizes ranging from 64 to 1024 characters and different overlap ratios. The processed token texts were stored as 768-dimensional vectors in a Pinecone database. Additionally, two n8n workflows were designed to implement the two main processes of token vectorizing and generative question answering. During the experiments, NotebookLM is introduced to generate 100 legal questions covering specific case features, legal concepts, and general inquiries. The results were categorized into emergent intelligence, true positives, false positives, and false negatives. The findings indicate that a chunk size of 128 characters without overlap yielded the best performance, effectively preserving context and providing longer, more accurate answers. Conversely, large chunk sizes or too much overlap caused information overload for the model, reducing query efficiency and accuracy. The study further compared token usage across different groups and found that the number of prompt tokens increased as the chunk size decreased, while the response length tended to stabilize under reasonable overlap settings. This suggests that the segmentation strategy has a direct impact on the performance of the RAG system. To ensure data security and prevent sensitive information leakage, the system adopts a local deployment strategy, combined with the highly flexible n8n workflow, effectively integrating the local language model with the vector database. This research not only validates the feasibility of applying RAG technology in the Chinese legal professional domain but also emphasizes the critical impact of text segmentation methods on answer quality. It provides important references for future expansion of judicial databases and enhancement of language models, aiming to promote the digital transformation of the intelligent judiciary and police investigation.

KEYWORDS- Large Language Models, Retrieval-Augmented Generation, Judicial Documents, Document Chunking

I. INTRODUCTION

The continuous and rapid development of LLMs can be traced back to the seminal 2017 paper "Attention Is All You Need" by Vaswani et al. [1]. Their work introduced the transformer architecture, a framework that adopts an attention mechanism instead of recurrent layers, enabling the model to capture global dependencies within the data. Subsequently, in 2019, Devlin et al. proposed Bidirectional Encoder Representations from Transformers (BERT), which significantly improved language processing performance through its novel bidirectional, unsupervised language representations [2]. The tremendous success of LLMs is attributed to their advanced architectures with billion-level parameters. From a technical perspective, the great success of LLMs owes to their sophisticated architectures, which are pretrained on massive training corpora from diverse sources, involving billions of parameters. These technological advancements have

enabled LLMs to demonstrate outstanding capabilities, especially in language understanding and generation, as well as contextual learning. However, despite these successes, LLMs still face inherent limitations, such as a lack of domain-specific knowledge, the problem of "hallucination," and the significant computational resources needed to update the LLMs [3][4]. These issues are particularly pronounced in specialized fields like medicine and law. Large language models (LLMs) have been applied to certain legal aspects and then constrained within a fixed database of court legal decisions. For example, in contract reviews, AI software can be used to review contracts, deliver recommendations, and even generate negotiation analysis, such as LITERA Kira and Luminance - Legal-Grade™ AI [5][6]. These intelligent contract review and clause analysis software can quickly identify potential risk clauses and contract inconsistencies for optimizing the contract review process. For case search, LLM combined with the search system can accelerate legal case searches to provide relevant judgment summaries and legal article comparisons in assisting legal professionals in document drafting, summarization, document analysis, and linked citations with conversational search software, such as Lexis+ AI and Westlaw Edge [7][8]. The legal AI software, however, is based on the country's legal decisions and fixed databases for lawyers to use.

Moreover, the challenges of overcoming the hallucination problem become even more difficult due to the substantial computational resources required for fine-tuning LLMs with domain-specific or the latest data. This significantly hinders the widespread adoption of LLMs in various domain-specific applications. Additionally, the most recent information is not included in the training data. To tackle these issues and reduce hallucinations, prompt engineering and contextual prompts have been developed, and fine-tuning for specific applications is sometimes performed to improve the accuracy of responses in particular domains. Fine-tuning involves high costs but incorporating prompts to establish scenarios and response templates can enhance the precision of replies. Furthermore, domain-specific terminology or customary usage can be further reinforced through fine-tuning.

To incorporate new data as a reference for LLM responses, research has been conducted on Retrieval-Augmented Generation (RAG). This technology aims to address the inherent limitations of LLMs, including the generation of hallucination information and the possibility that internal knowledge may be outdated-ed. RAG can provide reliable and up-to-date external knowledge, offering great convenience for numerous tasks. Especially in the era of AI-Generated Content (AIGC), the powerful retrieval capability to supply additional knowledge enables RAG to assist existing generative AI in producing high-quality outputs. Recently, LLMs have demonstrated revolutionary capabilities in language understanding and generation, though they still face inherent limitations such as hallucinations and outdated knowledge [9]. In 2024, Jimeno Yepes et al. investigated the impact of document chunk size on the efficiency and accuracy of information retrieval within RAG systems [10]. Their study analyzed 84 financial reports and utilized 150 authentic financial queries to evaluate the retrieval performance and response quality in the context of financial document analysis.

This paper aims to investigate the application of this technology to legal judgments in Taiwan, enabling generative artificial intelligence to perform case retrieval and produce responses that are specifically tailored to legal cases within a specialized domain. In other words, the proposed system is designed to emulate the reasoning processes articulated in prosecutors' indictments and judges' verdicts, as well as the evidence evaluation and judicial reasoning embedded in judges' rulings. This system has the potential to assist law enforcement officers in comprehending critical aspects of evidence collection, improving their understanding of prosecutorial arguments, and offering insights into judicial reasoning, as well as elucidating factors that contribute to the inaccessibility of certain information within the judicial system. Accordingly, the present study is structured around the fundamental framework of RAG, which consists of three core components: retrieval, augmentation, and generation.

Given the study's focus on a specific legal domain, 35 prosecutors' indictments and judges' verdicts were selected as input documents for the retriever component. A large language model (LLM) was employed to perform embedding for segmenting these original documents. Those segmentations become 768-dimensional vector token embeddings that were subsequently stored in a vector database. The vectors retrieved from this database served as augmentation inputs, which were then used by the generator to answer queries posed by law enforcement personnel. The primary objective of this research is to examine the factors of chunk size and overlap parameters during the embedding process for Traditional Chinese legal texts. Five chunk sizes, defined by character lengths of 64, 128, 256, 512, and 1024, were investigated under various overlap conditions of no overlap, one-quarter overlap, one-third overlap, and one-half overlap. This experimental design aims to elucidate the applicability of Retrieval-Augmented generation (RAG) technology to Taiwanese legal documents. Utilizing the Gemini language model for Traditional Chinese embeddings, this study investigates the following research questions: (1) To what extent does chunk size affect retrieval accuracy, and specifically, does a smaller chunk size enhance accuracy? (2) What is the impact of varying degrees of overlap on retrieval performance, particularly whether increased overlap leads to improved accuracy? (3) How can retrieval accuracy be most effectively quantified? Would it be based on the number of prompt tokens derived from chunked queries, the completeness of prompts retrieved from the vector database, or the length of generated responses?

The following section reviews related research pertinent to criminal and legal cases, emphasizing the key findings and contributions of prior studies. Section III details the experimental setup and methodology employed in this study. Section IV presents the experimental results, followed by a discussion in Section V. Finally, the concluding remarks and directions for future research are outlined.

II. RELATED WORKS

A. Retrieval-Augmented Generation

Large pre-trained language models have been demonstrated to encode factual knowledge within their parameters and attain state-of-the-art performance subsequent to fine-

tuning on downstream natural language processing tasks. Nonetheless, their capacity to acquire and accurately manipulate domain-specific knowledge remains constrained. Consequently, in knowledge-intensive tasks, the performance of these models is inferior to that of task-specific architectures, primarily due to the occurrence of potential hallucinations. Furthermore, generating robust justifications for the models' decisions and effectively updating their embedded world knowledge continue to represent significant open challenges within the research community. To address these limitations, Lewis et al. proposed the Retrieval-Augmented Generation (RAG) framework, which integrates pre-trained parametric memory with non-parametric memory to enhance language generation [11]. In this approach, the parametric memory is instantiated as a sequence-to-sequence model, whereas the non-parametric memory is implemented as a dense vector index constructed from Wikipedia.

In their study, the authors proposed employing a dual-encoder (bi-encoder) architecture for the Dense Passage Retriever (DPR). Specifically, the document encoder d(z), which is based on BERTBASE, produces dense vector representations of documents z, while a distinct query encoder q(x), also derived from BERTBASE, generates dense representations of queries x. Retrieval of the top-K documents is then performed using Maximum Inner Product Search (MIPS). Lewis et al. utilized a pre-trained sequence-to-sequence (seq2seq) model as the generator component. Specifically, they employed the BART-large model, comprising 400 million parameters, which is a pre-trained Transformer-based seq2seq architecture. The BART model is pre-trained using denoising objectives that incorporate various noise functions, demonstrating strong performance across a diverse array of generation tasks. The retriever identifies the top-K most relevant documents z based on the input query x. To facilitate the generator's utilization of the retrieved information, the simplest and most direct approach involves concatenating the input query x with the retrieved documents z, which collectively serve as input to the BART generation model. In this manner, the retrieved documents function as supplementary context to guide the generation process of the language model. The authors proposed two distinct models: the RAG-Sequence model and the RAG-Token model. The RAG-Sequence model maintains the same retrieved document as a fixed latent variable throughout the generation of the entire sequence. This latent variable is marginalized using a top-K approximation to estimate the conditional probability of the output sequence $p(y|x)$. In contrast, the RAG-Token model permits the selection of different latent documents for the generation of each target token, thereby enabling the generator to integrate information from multiple documents during the generation process. For each retrieved document, the generator produces a probability distribution over the subsequent output token, which is subsequently marginalized across all retrieved documents.

In 2024, Cuconasu et al. reported that, under certain conditions, the inclusion of randomly selected documents representing as a noise within prompts can enhance the accuracy of large language models (LLMs) by up to 35% , thereby challenging conventional assumptions regarding the necessity of document relevance in retrieval-augmented generation [12]. Cuconasu's study concentrated on Information Retrieval (IR) and classified documents into four categories for experimental evaluation: Gold Documents, Relevant Documents, Distracting Documents, and Random Documents. Gold Documents correspond to the original Wikipedia page snippets within the dataset that contain the correct answer and are directly pertinent to the query context. Relevant Documents constitute a subset of Gold Documents that both contain the correct answer and contribute to addressing the query. Distracting Documents are semantically similar to the query but lack the correct answer, serving as a means to assess the generator's capacity to discriminate between relevant and irrelevant information. Random Documents, which are completely unrelated to the query or answer, are used to assess the model's capability to handle entirely irrelevant information. They utilized the FAISS IndexFlatIP indexing system to process a corpus of approximately 21 million documents, followed by retrieval using a BERT-based dense retriever called Contriever, which was trained in an unsupervised manner using contrastive loss. The LLMs used in the experiments include Llama2, MPT, Phi-2, and Falcon. Among them, Llama2 can handle up to 4,096 tokens, while the other models are limited to 2,048 tokens. The study revealed that an increase in the number of distracting documents corresponded with a pronounced and gradual decline in the accuracy of large language models (LLMs). In certain instances, accuracy decreased by more than 0.38, representing a 67% reduction. Notably, the addition of a single distracting document resulted in a substantial accuracy decline, with a maximum observed decrease of 0.24 (25%). These findings suggest that LLMs tend to disproportionately attend to noisy documents, thereby neglecting the gold documents and generating erroneous outputs. This phenomenon constitutes a significant challenge for retrieval-augmented generation (RAG) systems, particularly in real-world information retrieval settings where semantically relevant but answer-absent documents are prevalent, potentially diverting LLMs from identifying the correct answers.

In our experiments, we observe that the noise phenomenon encompasses judicial documents from various categories, characterized by diverse case descriptions and developments. From a design perspective within the retrieval-augmented generation (RAG) framework, documents are segmented into chunks and stored in tokenized form. A Top-4 retrieval strategy is employed to select tokens, followed by marginalization, thereby enabling the natural language generation component to achieve more effective selection based on this model.

B. Criminal Investigation Analysis with RAG

Following advancements in technologies such as large language models (LLMs) and retrieval-augmented generation (RAG), Skipanes et al. introduced an application termed KriRAG, which is specifically designed to support criminal investigators in mitigating challenges related to information overload in 2025. [13] The KriRAG system is designed to enhance thematic processing in criminal investigations by integrating summarization and memory mechanisms within a retrieval-augmented generation (RAG) framework. The study employed three data sources: two highly confidential homicide case files from the Norwegian police (Case A and Case B) and one publicly accessible case. Information needs for the open case were manually annotated by criminal investigators, and final responses were provided by three investigators across all

cases, thereby establishing the ground truth. The raw data underwent filtering and validation to mitigate issues related to document scanning and OCR errors. Subsequently, a specialized sentence vector encoder, SBERT, trained on the Norwegian BERT model, was employed to generate 768-dimensional embedding vectors. These embeddings were stored and queried using the Chroma vector database, which facilitates the management and filtering of sentence-level embeddings. For retrieval, the Hierarchical Navigable Small World (HNSW) algorithm was employed to identify the k nearest neighbors, with similarity measured by the Euclidean distance metric. The language models evaluated in the KriRAG experiments included Gemma-2-9B, Gemma-2-27B, Gemma-2-9B-it, Llama-3.1-8B, and Phi-3-Medium.

The author formulates three research questions (RQs) to guide systematic evaluation. (1) RQ1. Document Retrieval: To what extent can KriRAG effectively identify and retrieve documents pertinent to the specified information needs? This is assessed using metrics such as Precision, Recall, F1 Score, and Mean Average Precision (MAP@kk). (2) RQ2. Relevance Assessment: To what extent can KriRAG accurately evaluate whether the retrieved information satisfies the information needs? This evaluation is conducted via manual assessment, employing criteria including relevance reasoning, fabrication of single or multiple details, contextual errors, and misunderstandings. (3) RQ3.To what extent do the answers generated by KriRAG align with those provided by investigators concerning the information needs, and how accurate are these responses? This is also assessed manually, with criteria focusing on reasoning and coherence.

III. SYSTEM DESIGN AND WORKFLOW IMPLEMENTATION

This section delineates the experimental framework. Initially, the design of the selected judicial documents is described, followed by an explanation of the implementation procedures. The implementation is comprised of two primary processes: the first involves vectorizing the documents through segmentation, while the second entails conducting queries based on simulated questions pertaining to the documents. These questions were designed and generated using NotebookLM.

A. Legal Case Data Ingestion

The input files for the experiment were obtained from the Judicial Yuan's judgment database, comprising a total of 30 selected documents. [15] These include 8 cases involving violations of the Telecommunications Management Act, 8 cases pertaining to breaches of the Personal Data Processing Act, 6 cases related to violations of the Assembly and Parade Law, 2 fraud cases, and one case each concerning money laundering, defamation, cyber intrusion, dereliction of duty, murder, and coercion. The judicial documents span 10 distinct crime categories, with 6 categories represented by a single document each, thereby intentionally augmenting what Cuconasu et al. characterize as "noise." Each document addresses different criminal charges, with varying emphasis on the descriptions of criminal facts and evidence depending on the crime type. The selection prioritizes cases with diverse characteristics, especially those related to the Telecommunications Management Act, the Personal Data Protection Act, and the Assembly and Parade Law. All selected documents consist of judicial verdicts issued within the past two years.

B. n8n Orchestration Workflows

n8n is an open-source, versatile automation workflow platform that provides an intuitive visual interface enabling users to construct diverse workflows through drag-and-drop node assembly, thereby obviating the need for extensive coding. The platform includes built-in connectors for over 200 widely used application services, such as Google Sheets, Gmail, and Slack, and supports integration with custom APIs via HTTP request nodes, offering substantial extensibility. Additionally, n8n incorporates features such as conditional branching, loops, and custom code nodes, facilitating the creation of highly customized workflows. Utilizing n8n enables users to improve operational efficiency and reduce the incidence of human error, making the platform accessible to individuals with varying levels of technical expertise. Moreover, its support for self-hosted deployment ensures compliance with data privacy and security requirements by allowing sensitive information to remain under user control. In this study investigating chunk size in retrieval-augmented generation (RAG) technology, n8n was selected as the platform for workflow design. Section III-B1 details the development of a workflow for vectorizing judicial documents and storing the resulting embeddings in a database, whereas Section III-B2 describes the creation of a generative workflow enabling users to query this database.

1) Vectorization Flow

To create a document processing workflow that tokenizes texts with varying chunk sizes and stores them in the Pinecone vector database, an n8n-based workflow was developed, as illustrated in Figure 1. This workflow facilitates the tokenization of judicial documents and subsequent storage within a vector database. Judicial documents, stored in PDF format on a cloud drive, are sequentially retrieved through a looping mechanism. Upon downloading, a JavaScript routine removes line breaks and extraneous symbols to convert the PDFs into plain text. The resulting text is then processed using Google Gemini's large language model (LLM) and segmented into sentences employing a recursive character text splitter, following the methodology proposed by Esmaeil Narimissa et al. in 2024 [14]. Within the recursive character text splitter, parameters for chunk size and chunk overlap are systematically defined. Chunk sizes are categorized into five principal values: 64, 128, 256, 512, and 1024 characters. The overlap ratios are set as none, one-quarter (1/4), one-third (1/3), or one-half (1/2). This experiment conducts artificial intelligence experiments on court judgments by applying a chunking approach to legal texts. A total of ten categories of judicial decisions were selected for analysis, covering a diverse range of criminal and regulatory domains. The dataset includes telecommunications and broadcasting (8 cases), personal data protection and document forgery (8 cases), assembly and parade law violations (6 cases), fraud (2 cases), money laundering (1 case), defamation (1 case), cyber intrusion (1 case), malfeasance in office (1 case), homicide (1 case), and coercion (1 case). These judgments were segmented into smaller textual units (chunks) to facilitate downstream processing. Let c^t denote the t-th category of judicial decisions, where t indexes distinct

categories. τ denotes the chunk size and $o \in \{\frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1\}$ denotes the overlap ratio. The chunking function that segments the document into a set of chunks:

$$C = chunk(c^t, \tau, o) \quad (1)$$

In cases where the calculated overlap results in a non-integer value, the result is truncated, as character counts must be represented by integers. For example, when the chunk size is 64 and an overlap ratio of one-third is applied, the chunk overlap is computed as $\lfloor 64/3 \rfloor$. The configuration method for these approaches is shown in Figure 2. Following this tokenization process, each chunk is converted into a 768-dimensional embedding vector, which is subsequently stored in the corresponding vector database. Consequently, 16 vector databases are created. Each identified by a specific combination of chunk size and overlap ratio. Overlap ratios are denoted by the symbols *z, q, t,* and *h*, corresponding respectively to no overlap, one-quarter, one-third, and one-half. For instance, a vector database generated with a chunk size of 1024 and an overlap ratio of one-half is labeled as overlap-h-1024.

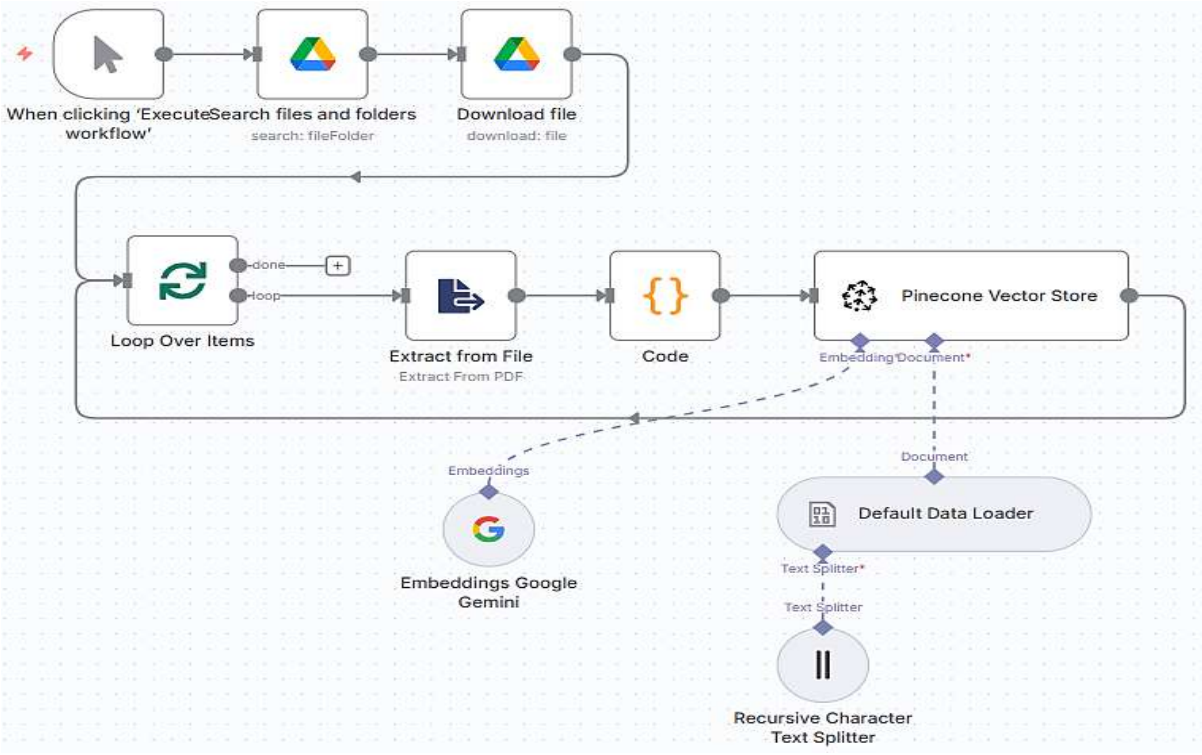


Figure 1: Vectorization Flow

The resulting chunks are then processed using the Google Gemini embedding model for semantic analysis and vectorization and subsequently stored in the Pinecone vector database. The function is as following:

$$\varepsilon = \{E(c_i^t) | c_i^t \in C\} \quad (2)$$



Figure 2: Settings of the Recursive Character Text Splitter

2) Question Query Flow

Following the processing of judicial documents described in the preceding section, sixteen vector databases were constructed using varying chunk sizes. These vector databases were subsequently integrated into the workflow depicted in Figure 3. The configuration method of each functional block in Figure 3 is shown in Figure 5. The experimental procedure entailed storing 100 inquiry questions on a local hard drive, which were then sequentially submitted to an AI Agent for question answering. The generative large language model employed by the AI Agent connected to local LM Studio software shown in Figure 4, utilizing credentials configured through the Open Chat Model function, with the meta-llama-3.1-8b language model serving as the underlying LLM. Within the Simple Memory component, a window size of five was configured to retain the last five responses, thereby enriching subsequent replies. The sixteen vector databases from the prior section were individually selected within the Pinecone vector store for experimental evaluation. For the AI Agent's prompt configuration, no contextual scenarios were established but a fixed prompt was applied: "You are a criminology expert; if relevant data exists in the database, please respond accordingly, and answer in Traditional Chinese." This approach minimizes experimental variables. Furthermore, queries to the Pinecone vector store were consistently limited to the top four results, further controlling potential variable influences.

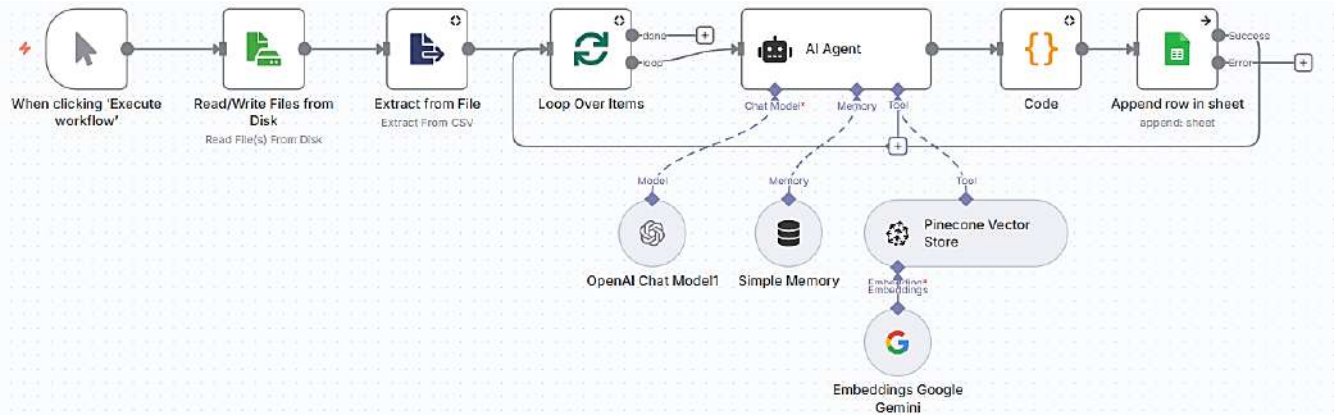


Figure 3: Question Query Flow.

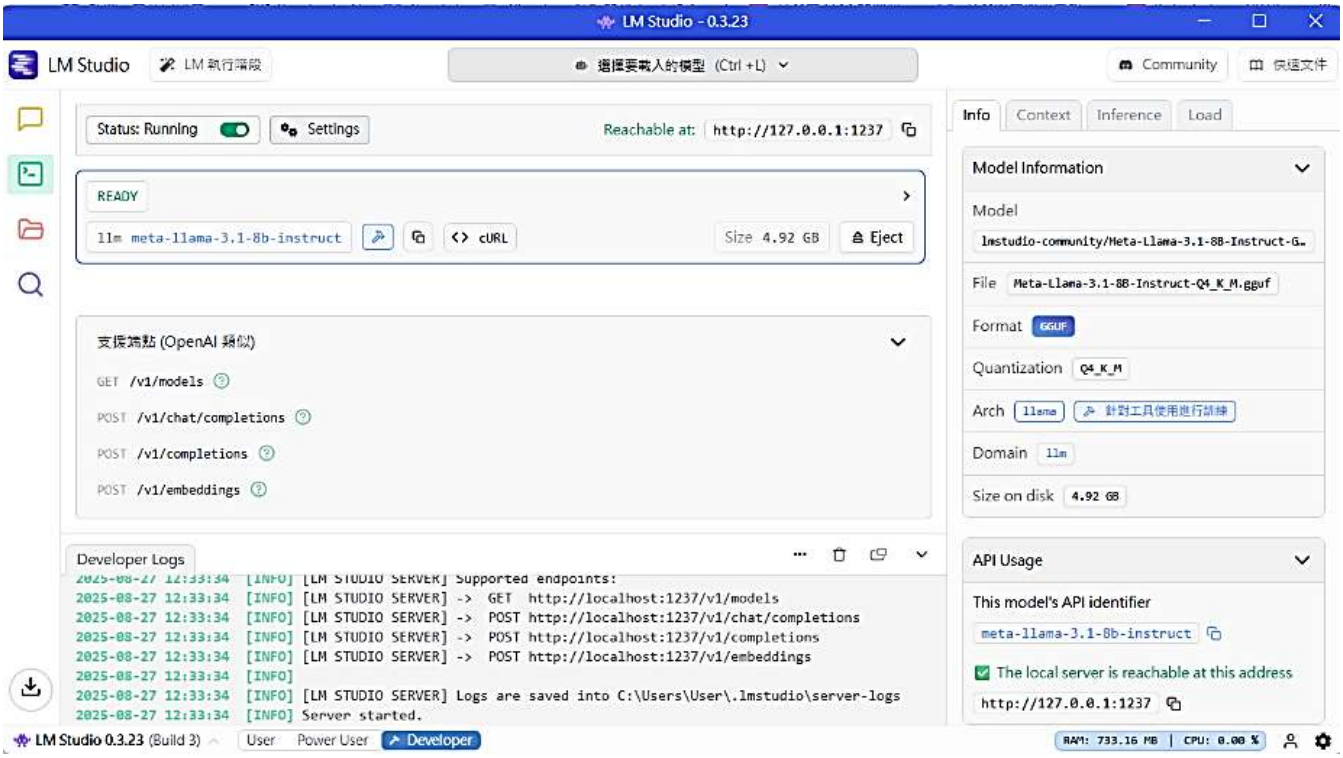


Figure 4: Large language model on the local device.

C. Generative Question Formulation

The 100 input questions mentioned in the previous section were generated by uploading 30 judicial documents mentioned in Section III-A to Google NotebookLM, which produced 100 questions. These 100 questions can be classified into 3 categories. To evaluate the performance of the proposed RAG system, we categorize the queries into three distinct types:

- Fact-Based Questions Q_{FB}: These questions have explicit and verifiable answers grounded in the source

documents, often involving specific entities such as personal names or company names. The system is expected to generate precise and unambiguous responses.

- Legal Reasoning Questions Q_{LR}: These questions focus on legal concepts and statutory provisions referenced in the judgments. They are designed to assess the model's ability to generate coherent legal explanations while leveraging the RAG framework to reduce hallucinations through grounded retrieval.
- Open-Ended Questions Q_{OE}: These questions are not strictly confined to the content of the given judgments and allow the system to produce broader legal knowledge and reasoning. This category evaluates the system's capacity to generate contextually relevant and knowledge-enriched responses beyond the retrieved documents.

The first category consists of fact-based questions that involve specific entities mentioned in the documents, such as names of individuals or companies. The second category consists of questions related to legal concepts and statutory interpretation. For example, one question asks: "The judgment distinguishes the boundaries between accomplices and co-perpetrators. Why were the defendants' actions identified as aiding in fraud rather than being co-perpetrators, and what is the distinguishing standard?" The third category consists of questions that do not fall into the first two categories and are more open-ended in nature. For instance, one question asks: "Despite some defendants spraying chili spray, throwing objects, or seizing iron bars, why did the court determine that these actions did not constitute obstruction of a lawful assembly under the specific circumstances?" The distribution of these question types is 20%, 60%, and 20%, respectively. The responses of the query questions are noted as follows:

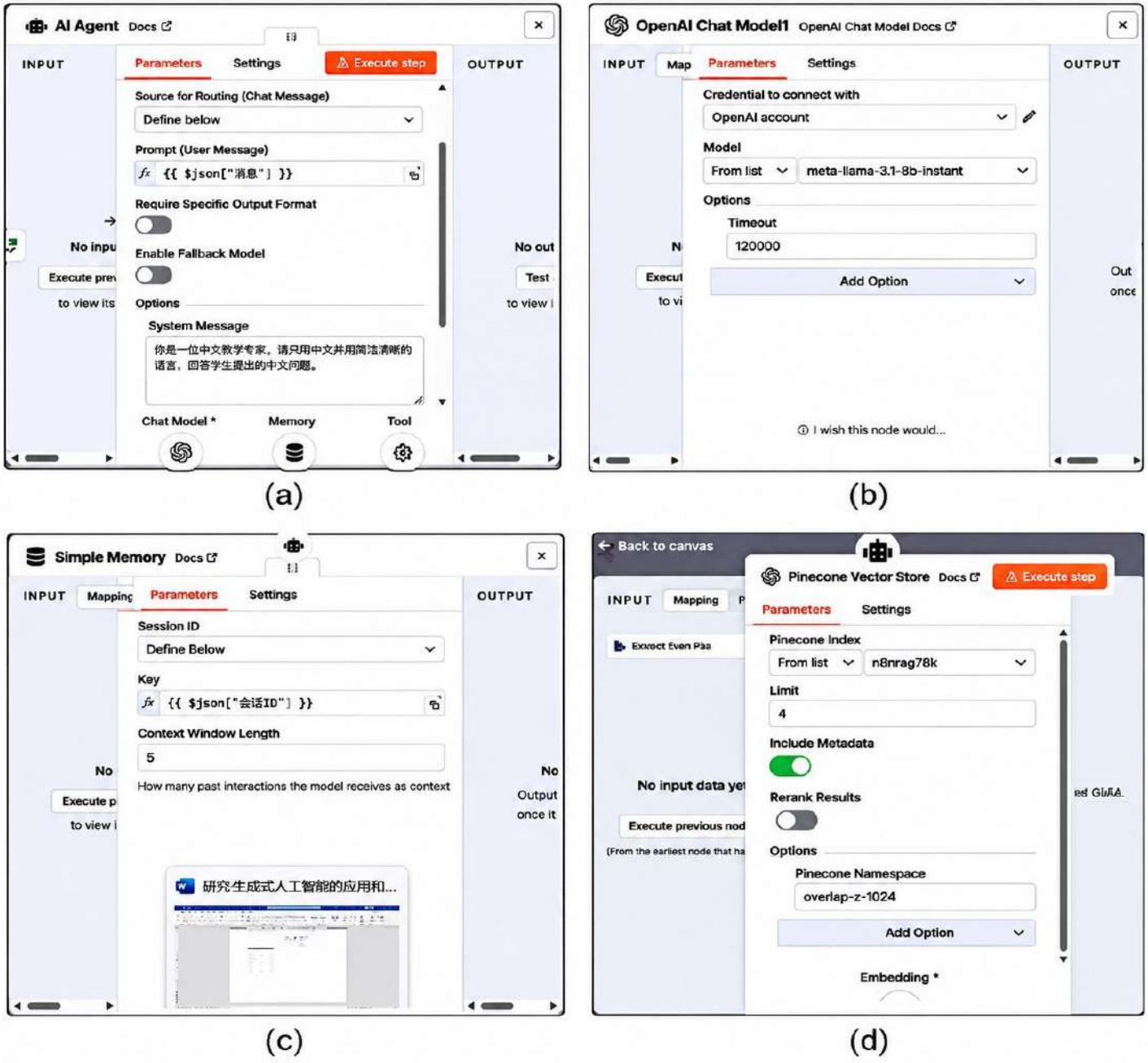


Figure 5: Settings in the Question Query Flow: (a) AI Agent prompt; (b) Open Chat Model; (c) Simple Memory with a five-response history window; (d) Pinecone Vector Store

$$R = R_{q_i \in \{Q_{FB} \cup Q_{LR} \cup Q_{OE}\}}(q_i, \varepsilon, k) \quad (3)$$

where k is the k-th most relevant chunk.

We define a retrieval and response function $R = R_{q_i \in \{Q_{FB} \cup Q_{LR} \cup Q_{OE}\}}(q_i, \varepsilon, k)$represents a query drawn from fact-based, legal reasoning, and open-ended question sets, ε denotes the embedding model, and k indicates the number of retrieved results. The outputs of R are categorized into three types:

$$R: (q_i, \varepsilon, k) \rightarrow \{TP, FP, FN\} \quad (4)$$

We define a true positive (*TP*) as the case where the RAG vector database is correctly queried, confirming that the chunked text has been successfully imported and indexed. Conversely, a false positive (*FP*) indicates a case where a valid answer should have been returned because the question was explicitly designed based on chunked data in the vector database. Finally, a false negative (*FN*) refers to a case where the quality of the resulting response is low. Some results in *TP* may also be in *FN*. Within the *TP* category, we further identify a subset of responses termed Emergent Responses (*ER*), which primarily occur in open-ended queries (Q_{OE}). *ER* refers to cases where the system retrieves semantically related information and generates meaningful and satisfactory answers that extend beyond explicit matches in the database.

IV. EXPERIMENTAL RESULTS

This section is organized into two parts to present the experimental results reported in the previous section. Section IV-A examines the effects of varying chunk sizes, while Section IV-B presents the outcomes of the Question Query Flow.

A. Impact of Chunking Size on Token Usage

In evaluating the experimental results concerning Traditional Chinese legal documents, the study aimed to assess the impact of chunk size on retrieval-augmented generation (RAG) performance using three objective metrics for querying the vector database, which are Prompt Tokens, Completion Tokens, and Response Length. Prompt Tokens denote the number of tokens input into the query and the referenced vector database after processing by the large language model (LLM). Completion Tokens refer to the number of tokens generated following data retrieval and matching by the vector database. Response Length represents the length of the final generated response. Initially, chunk sizes of 1024, 512, 256, 128, and 64 were compared under conditions of no overlap. As depicted in Figure 6, the number of Prompt Tokens tended to increase as the chunk size decreased. The chunk size of 512 exhibited the greatest number of outlier anomalies. Further analysis revealed that these anomalies predominantly resulted from the presence of multiple unique Chinese bracket symbols within the questions. The questions, generated by NotebookLM, incorporated various bracket types such as 「」, 『』, 《》, and 【】, which influenced character segmentation and subsequently affected token counts. The generated text length with a chunk size of 64 produces longer responses. The median response lengths for chunk sizes of 1024, 512, and 256 are approximately 500 tokens, whereas chunk sizes of 128 and 64 yield higher median values.

Under a one-quarter overlap condition, the number of prompt tokens generated for the 1024 chunk size exhibits greater instability and a broader distribution range. A similar pattern is observed across experiments with varying overlap ratios, indicating that selecting longer chunk sizes with partial overlap results in less stable token counts. This variability suggests that larger chunks may encompass a broader range of content and potentially include more meaningful tokens. Notably, the median token counts for chunk sizes of 1024, 512, and 256 under these conditions approximate 1000. For chunk sizes of 128 and 64, the median token count decreases to approximately 500.

When the overlap is set to one-third, the number of Prompt Tokens generated for the 1024 chunk size remains less stable with a wider distribution. Under this condition, the median token counts for chunk sizes of 1024, 512, and 256 cluster around 500, whereas for chunk sizes of 128 and 64, the median token counts increase to about 1000.

Under a one-half overlap condition, the number of Prompt Tokens generated for the 1024 chunk size continues to exhibit instability and a wider distribution. However, the median Prompt Token values for all chunk sizes of 1024, 512, 256, 128, and 64 converge around 500. Completion Tokens, which indicate the number of tokens in the model's generated response, remain stable regardless of the chunk size or overlap ratio used. In terms of Response Length, measured by the character count of the output, experiments without overlap demonstrate that shorter responses tend to exhibit greater variability. In contrast, experiments with overlap ratios of one-quarter, one-third, and one-half show response lengths that are consistently short and closely clustered, indicating more uniformity.

B. Performance of Query Question Generation

After analyzing the token counts, this section examines the actual content of the generated responses. Table 1 displays the generative response results for the 100 questions, which are categorized into three types that are fully related questions containing special feature nouns from the documents, conceptual questions, and general questions. The distribution of question categories is detailed in Section III-C. The corresponding databases store 768-dimensional token vectors generated through the chunking process applied to the 30 judicial documents, as previously described.

Table 1: Comparison of generative response results by chunking configuration

Database	Emergence	TP	FP	FN
z-1024	19	86	0	14
z-512	19	92	1	7
z-256	19	91	0	10
z-128	20	96	1	5
z-64	16	89	2	11
q-1024	8	38	0	63
q-512	18	88	0	13
q-256	15	84	1	14
q-128	19	93	1	8
q-64	17	89	1	13

t-1024	8	30	0	66
t-512	17	91	0	9
t-256	18	94	0	13
t-128	16	90	0	12
t-64	19	96	0	7
h-1024	6	30	1	69
h-512	12	65	1	34
h-256	17	93	0	14
h-128	18	93	0	13
h-64	18	91	0	11

z: no overlap; q: 1/4 overlap; t: 1/3 overlap; h: 1/2 overlap. Emergence is a subset of TP.

The generative response results are categorized into four types which are Emergence, *TP*, *FP*, and *FN*. In Table 1, it is clear that apart from some databases obtained from the different chunking policy, which did not yield good results, most results demonstrate the successful effects of large language models. Emergence refers to instances where generated results reference information from similar cases in producing meaningful and satisfactory outputs for conceptual and general questions. *TP* denotes queries that correctly target the vector database in offering descriptive responses based on the database for fully related questions or indicating the absence of relevant data for conceptual and general questions. *FP* describe cases where responses claim no clear data is found in the database for fully related questions but nonetheless provide incorrect or misleading answers. In Table 1, the *FP* numerical representation language model provided meaningless responses to these 60 questions that were entirely relevant to the chunked judgment. The responses should have referred to the RAG vector database. That indicates the poor performance of the vector database during LLM data retrieving in the chunk size. *FN* refers to instances where the model either fails to respond or generates an excessively short reply that merely repeats the question. Because large text chunks introduce excessive text context into the embeddings, retrieving from the vector database often yields sub-optimal results. This leads to poor response quality, characterized by either incorrect information or redundant outputs that simply repeat the question.

In this experiment, when a large number of tokens were inputted and generated, the error rate in TP rose significantly. For instance, in the cases of q-1024, t-1024, h-1024, and h-512, this likely resulted from the large language models becoming overwhelmed by the volume of information, impairing their ability to accurately query the database. Additionally, in FP category, although there were 60 questions with clear targets, the number of corresponding responses was remarkably low, ranging only between 0 and 2. Furthermore, because FP shows only few in these 60 questions without gaining reference data from the database, the TP values implied these instances of conceptual and general questions where no data was queried from the database. Thus, for conceptual and general questions that should have been generated the answers without referencing the database by relying solely on the large language model. As reflected in the higher numbers of prompt tokens input in these q-1024, t-1024, h-1024, and h-512 databases, the experiment results present that likely caused confusion in the large language model in overwhelming. Consequently, the model failed to generate answers effectively, as evidenced by the elevated FN counts in these chunking databases

V. DISCUSSION

This section discusses the research design and experimental results. The experiment involved 30 cases drawn from 10 different types of legal documents. These documents contain unique and diverse content, encompassing statements of criminal facts, types of evidence, and referenced laws. Based on these 30 legal cases, NotebookLM was employed to generate 100 questions, of which 60 were fully related to specific case details, 20 were legal conceptual questions, and the remaining 20 were general questions. This paper primarily investigates the effects of varying chunk sizes and overlap ratios in Retrieval-Augmented Generation (RAG) technology during the construction of a vector database. The objective is to develop an expert system for legal dialogues, intended to assist police officers in inquiries and discussions, thereby improving their operational efficiency. Given that police officers in Taiwan serve as frontline law enforcement personnel, they frequently engage in on-site investigations and evidence collection. After gathering evidence, officers transfer the information to prosecutors. The quality of these transfer reports, prepared by police personnel and submitted to prosecutors, is crucial. Additionally, police officers are sometimes required to explain these reports to prosecutors in face-to-face interactions. Furthermore, judges independently adjudicate cases, and police officers do not have direct access to judges. This creates a challenge for police officers in understanding how judges formulate and assess their convictions. Therefore, this study aims to leverage RAG technology to integrate prosecutors' indictments and judges' verdicts into vector databases. This integration enables police officers to interact with the vector databases as if consulting an expert system composed of judges and prosecutors, thereby facilitating better comprehension and decision-making.

The challenge lies in effectively vectorizing legal documents, specifically judgments and indictments, for storage in the database. This study investigates 20 different methods for processing these documents. Through this preliminary research, it was observed that Taiwanese legal documents denote proper nouns using brackets. However, the presence of various styles of brackets within these documents may interfere with the large language models' ability to accurately segment the documents and generate tokens. The impact of embedding chunk size varies depending on the analytical capabilities of large language models, and the character-cutting strategies employed by recursive text splitters are influenced by the characteristics of the text content. In the experiment, judicial documents were processed with chunk sizes of 1024, 512, 256, 128, and 64 tokens. If the chunk size is set too small, it fails to preserve the semantic integrity of words, sentences, and contextual information. The experimental results indicate that the chunking method using 128-character segments generated better response outcomes for these judicial documents. To address potential loss of word order and semantics during chunking, this study tested four overlap

lengths, which are no overlap, 1/4 overlap, 1/3 overlap, and 1/2 overlap. The findings revealed that the no-overlap method yielded the best results. This is primarily attributed to the capability of the current recursive text splitter, which performs cuts at punctuation marks, thereby minimizing errors in word segmentation.

The performance evaluation of this experiment employs two methods. The first method is quantitative, focusing on the number of tokens generated by the prompt. This includes tokens retrieved from the search vector database as well as the complete prompt tokens in the query vector database, with response length used to represent the quantitative data. The influence of different chunking sizes and overlap ratios on these quantitative metrics is illustrated in Figure 6, Figure 7, Figure 8 and Figure 9.

The second evaluation method is qualitative, based on manual assessment, which involves classifying the questions and judging the correctness and reference value of the generated answers, specifically evaluating their knowledge content.

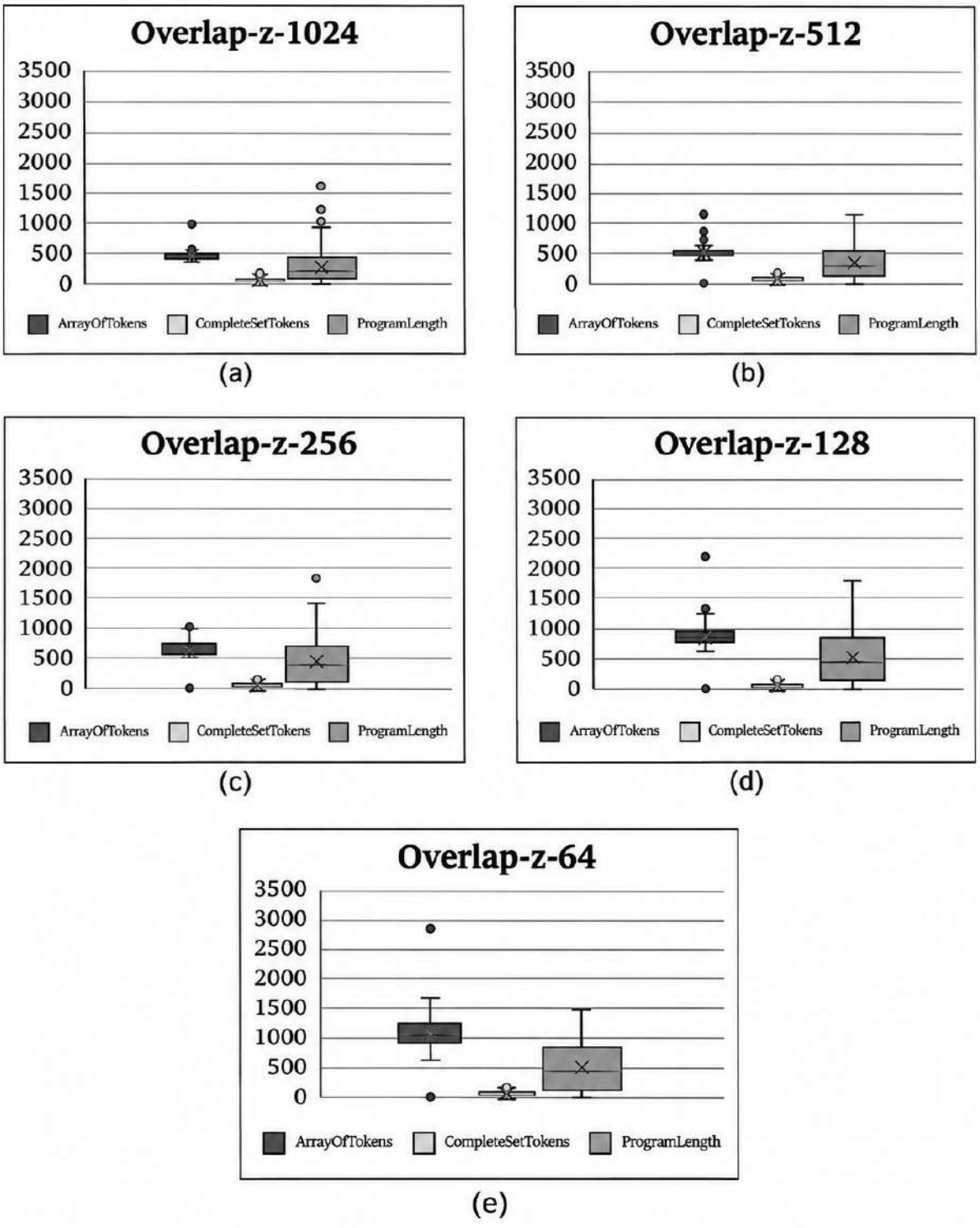


Figure 6: Token usage and response length with no overlap: (a) 1024, (b) 512, (c) 256, (d) 128, and (e) 64 characters per chunk

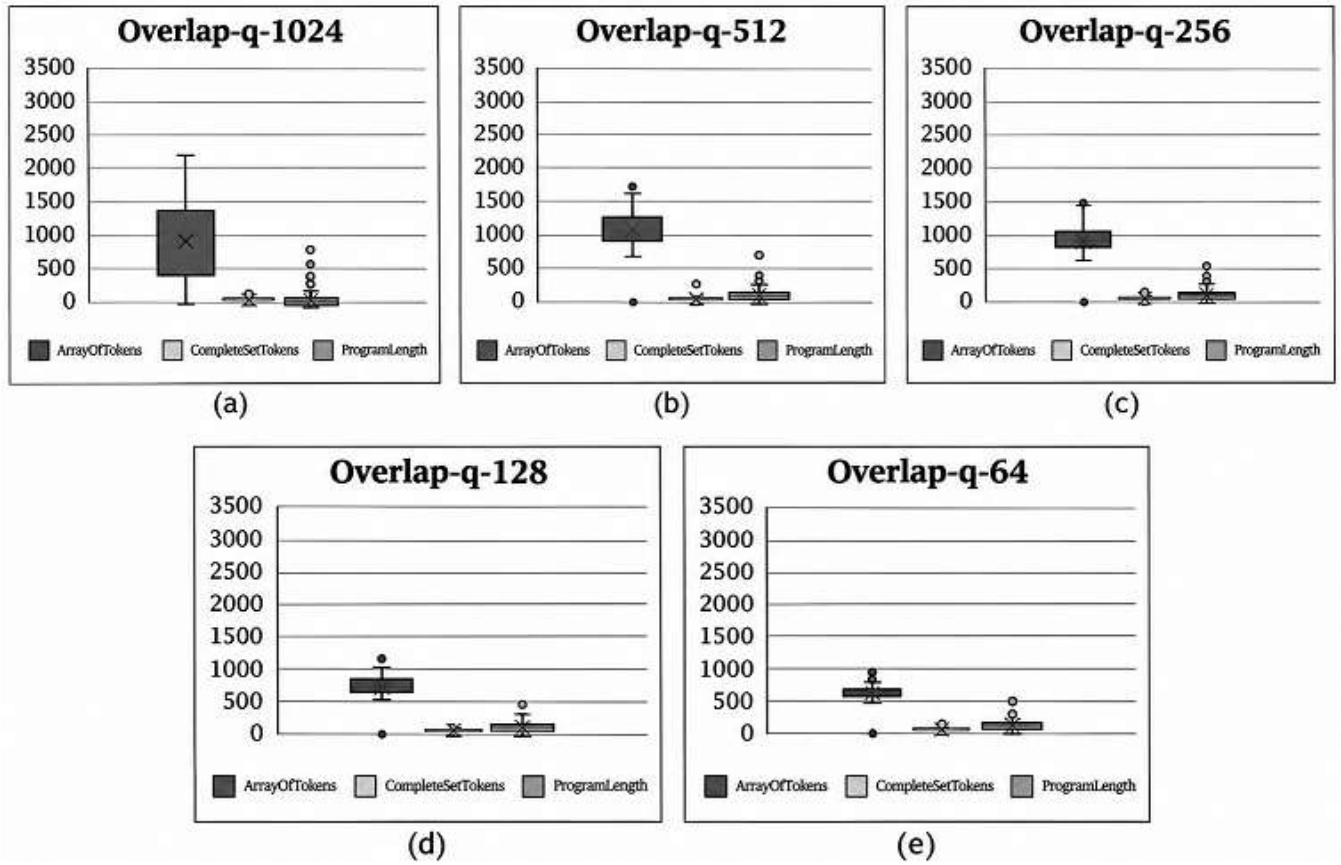


Figure 7: Token usage and response length with one-quarter overlap: (a) 1024, (b) 512, (c) 256, (d) 128, and (e) 64 characters per chunk

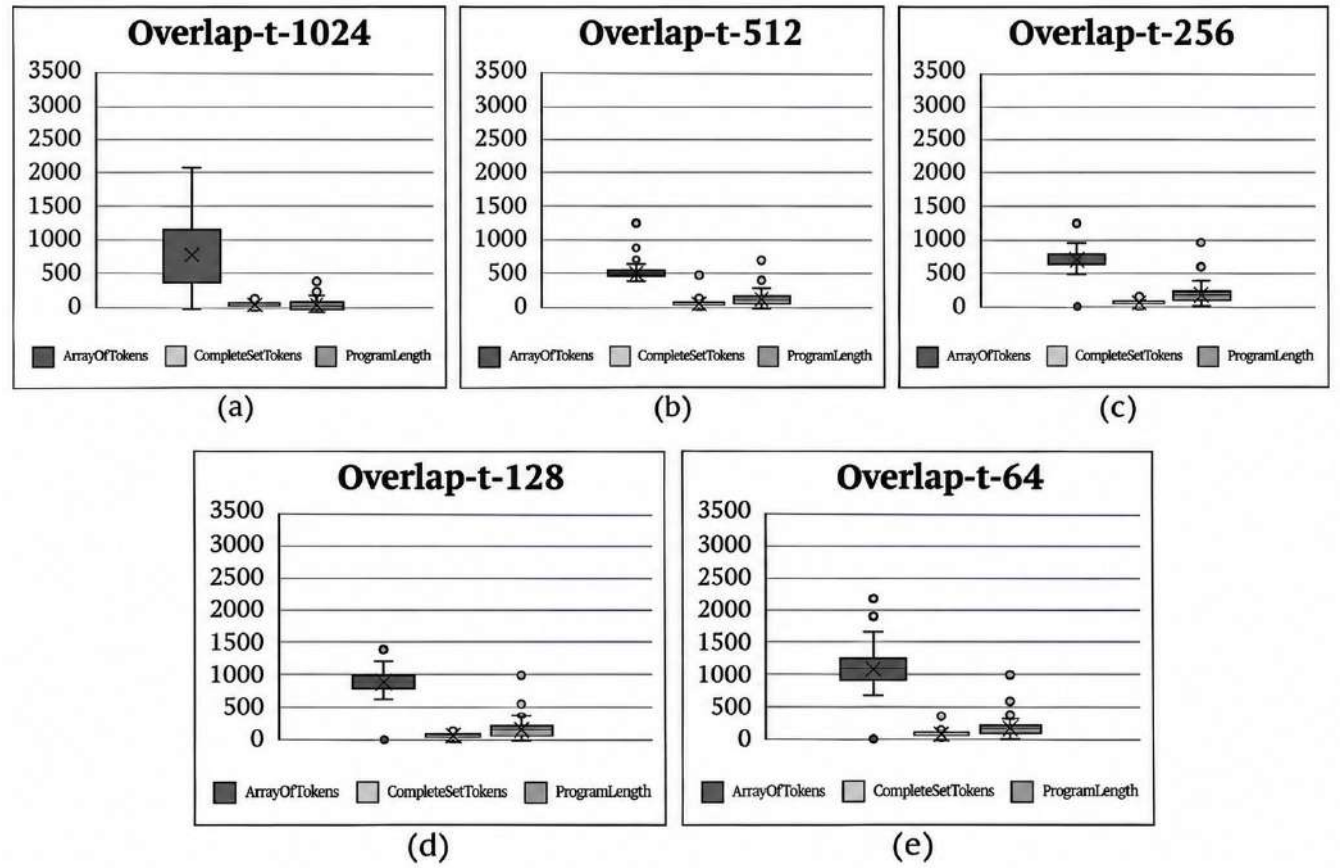


Figure 8: Token usage and response length with one-third overlap: (a) 1024, (b) 512, (c) 256, (d) 128, and (e) 64 characters per chunk

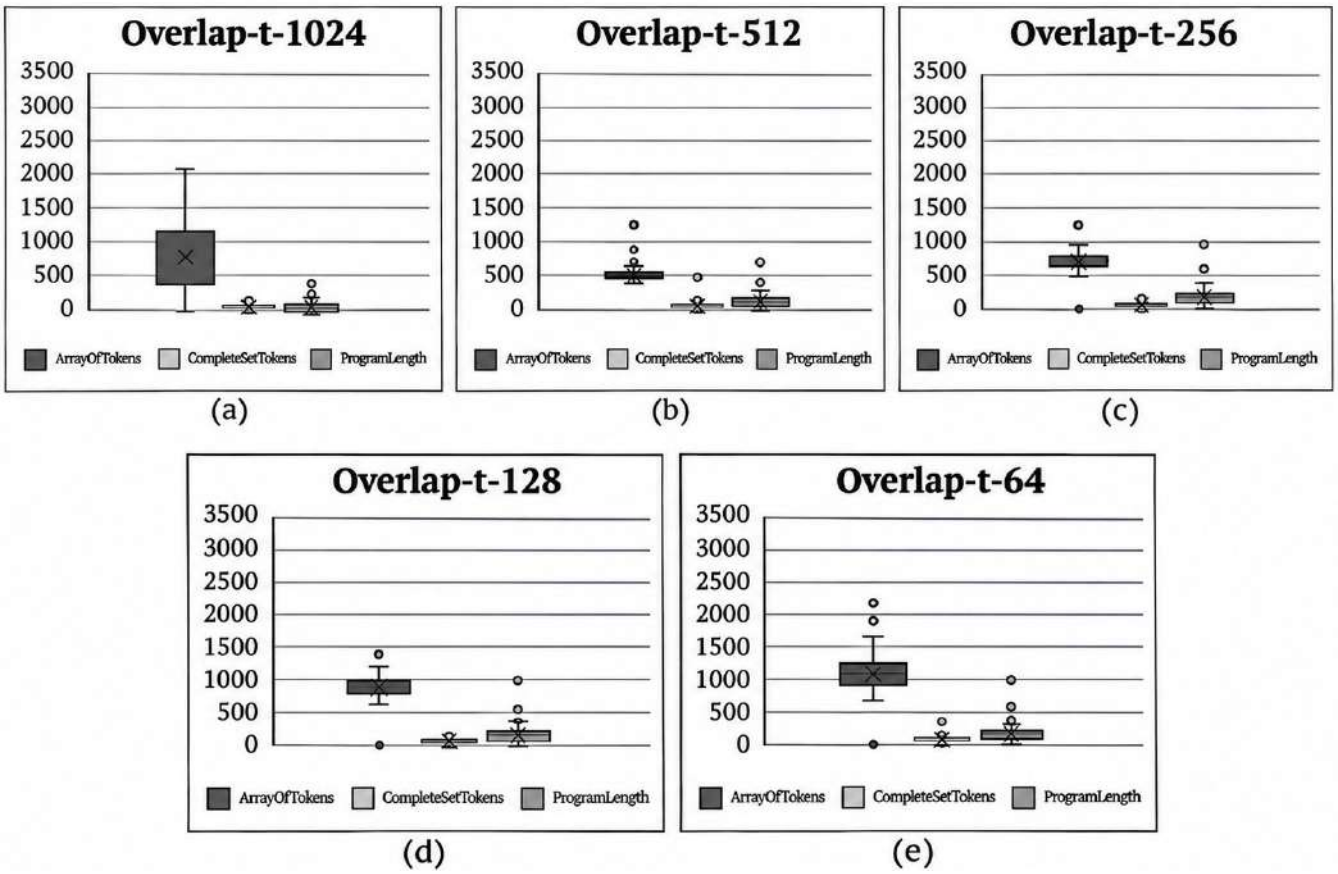


Figure 9: Token usage and response length with one-half overlap: (a) 1024, (b) 512, (c) 256, (d) 128, and (e) 64 characters per chunk

VI. CONCLUSIONS AND FUTURE WORKS

Large language models have increasingly gained prominence, concurrently giving rise to novel categories of information security incidents. For instance, in March 2025, the rapidly emerging Chinese artificial intelligence platform, DeepSeek, experienced a significant cyberattack that resulted in the inadvertent exposure of its internal database to the public internet. Specifically, DeepSeek's ClickHouse database was misconfigured, leading to two instances being accessible without authentication. These exposed databases contained over one million user chat logs in plaintext, along with API keys, system credentials, and other sensitive information. This security lapse permitted unauthorized actors to freely query the database and potentially misuse access rights, thereby underscoring critical vulnerabilities in the management of AI-driven information systems. [16] The internal large language model (LLM) system developed for this research is built upon LM Studio to mitigate the risk of information leakage concerning police-officers' queried and utilized data. The research framework utilizes n8n's workflow automation, comprising two distinct workflows: the Vectorization Flow and the Question Query Flow. These workflows facilitate the experimental process and the collection of results. Given that the experiments are conducted within the context of LLM applications, variability in outcomes may occur across trials. Furthermore, this study focuses primarily on analyzing chunking size and overlap sampling, and thus, factors such as hardware input capacity and the volume of judicial documents have not been exhaustively considered. For practical implementation and to develop a robust system suitable for police officers, the framework would require expansion to include more categories and a larger corpus of judicial documents for vectorization, stored within a more extensive vector database, and leveraging larger language models to achieve enhanced and more stable performance.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in Advances in Neural Information Processing Systems, vol. 30, 2017. Available from: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. 2019 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019. Available from: https://aclanthology.org/N19-1423/

[3] W. Fan, Z. Zhao, J. Li, Y. Liu, X. Mei, Y. Wang, J. Tang, and Q. Li, "Recommender systems in the era of large language models (LLMs)," 2023. Available from: https://arxiv.org/html/2307.02046v4

[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Advances in Neural

Information Processing Systems, vol. 33, 2020. Available from: https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230-Abstract.html

[5] "Contract reviews powered by AI," Litera. Available from: https://www.litera.com/products/kira/

[6] "Wherever computer meets contract," Luminance. Available from: https://www.luminance.com/ (accessed Sep. 10, 2025).

[7] "What is Lexis+ AI?," LexisNexis. Available from: https://www.lexisnexis.com/zh-tw/products/lexis-ai/

[8] "Expedite your most complex legal research tasks," Thomson Reuters. Available from: https://legal.thomsonreuters.com/en/products/westlaw-edge/

[9] W. Fan, Y. Ding, L. Ning, S. Wang, H. Li, D. Yin, T.-S. Chua, and Q. Li, "A survey on RAG meeting LLMs: Towards retrieval-augmented large language models," in Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining, 2024. Available from: https://dl.acm.org/doi/abs/10.1145/3637528.3671470

[10] J. Yepes, Y. You, J. Milczek, S. Laverde, and L. Li, "Financial report chunking for effective retrieval augmented generation," 2024. Available from: https://arxiv.org/abs/2402.05131

[11] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Advances in Neural Information Processing Systems, vol. 33, 2020. Available from: https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230-Abstract.html

[12] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonellotto, and F. Silvestri, "The power of noise: Redefining retrieval for RAG systems," in Proc. 47th Int. ACM SIGIR Conf. Research and Development in Information Retrieval, 2024. Available from: https://dl.acm.org/doi/abs/10.1145/3626772.3657834

[13] M. Skipanes, T. E. Jørgensen, K. Porter, G. Demartini, and S. Y. Yayilgan, "Enhancing criminal investigation analysis with summarization and memory-based retrieval-augmented generation: A comprehensive evaluation of real case data," in Proc. 31st Int. Conf. Computational Linguistics, 2025. Available from: https://aclanthology.org/2025.coling-main.334/

[14] Narimissa and D. Raithel, "Exploring information retrieval landscapes: An investigation of novel evaluation techniques and comparative document splitting methods," 2024. Available from: https://arxiv.org/abs/2409.08479

[15] "Judicial Yuan Judgment Query System," Judicial Yuan, Taiwan. Available from: https://judgment.judicial.gov.tw/FJUD/default.aspx

[16] "DeepSeek cyber attack: Timeline, impact, and lessons learned," CM-Alliance. Available from: https://www.cm-alliance.com/cybersecurity-blog/deepseek-cyber-attack-timeline-impact-and-lessons-learned/