CAS2105 Homework 6: Mini AI Pipeline Project 🤗

Heeji Kang (2018113009)

1 Introduction

This project focuses on building a **Retrieval-Augmented Generation (RAG)** pipeline [1] to solve Korean Criminal Law multiple-choice questions. Legal QA is a challenging domain because it requires precise grounding in specific laws and precedents, making it unsuitable for hallucination-prone language models acting alone.

The goal of this project is to demonstrate how a well-engineered "System" (consisting of Retrieval, Few-Shot Prompting, and Chain-of-Thought reasoning) can significantly boost the performance of a smaller, cost-efficient model like `gpt-4o-mini`. While large state-of-the-art models might solve these problems with internal knowledge, optimizing smaller models for specific domains is practically valuable for reducing deployment costs.

A RAG pipeline is implemented to retrieve relevant legal precedents and reason step-by-step to predict the correct answer (A, B, C, or D). This approach is compared against a naïve baseline, and the impact of system engineering on accuracy is analyzed.

2 Task Definition

- **Task description:** Answer multiple-choice questions about Korean Criminal Law. Each question provides four options (A, B, C, D).
- **Motivation:** Legal reasoning requires strict adherence to statutes and case law. A RAG system allows the model to "look up" the exact legal text before answering, minimizing errors and hallucinations.
- **Input / Output:**
  - Input: A question string and four option strings.
  - Output: A single character label matching the correct option {A, B, C, D}.
- **Success criteria:** Accuracy on the held-out test dataset (percentage of correct predictions).

3 Methods

3.1 Naïve Baseline

A **Random Baseline** was implemented to establish a lower bound for performance.

Baseline Implementation

- **Method description:** The system randomly selects one of the four options {A, B, C, D} with equal probability.

- **Why naïve:** It requires no learning, no understanding of the text, and no external knowledge. It represents pure chance.
- **Likely failure modes:** It is expected to fail 75% of the time statistically. It cannot distinguish between easy and hard questions.

3.2 AI Pipeline

A **Few-Shot RAG Pipeline** was designed leveraging OpenAI's `text-embedding-3-small` for retrieval and `gpt-4o-mini` for generation.

Pipeline Implementation

- **Models used:**
  - Retrieval: `text-embedding-3-small` (OpenAI)
  - Generation: `gpt-4o-mini` (OpenAI)
- **Pipeline stages:**
  1. **Indexing**: A knowledge base is constructed from the training set, where each entry contains the Question, Options, and Correct Answer (as the precedent). These entries are embedded using the embedding model to build a k-NN index.
  2. **Retrieval**: For a new test question, the query is embedded, and the Top-K ($K = 5$) most similar examples are retrieved from the knowledge base using Cosine Similarity.
  3. **Generation**: A prompt is constructed containing:
     - **Rules**: Strict instructions to use only retrieved context.
     - **Few-Shot Example**: A clear demonstration of the desired reasoning format.
     - **Context**: The retrieved Top-5 legal texts.
     - **Question**: The actual test question.

     The model generates a **Chain of Thought (CoT)** explanation followed by the final answer.
- **Design choices and justification:**
  - **Few-Shot + CoT**: Preliminary tests indicated that `gpt-4o-mini` struggled to reason zero-shot. Adding an example and enforcing step-by-step reasoning significantly improved adherence to the retrieved context.
  - **Top-K=5**: Initially, Top-K=10 was tested, but it was found that excessive context introduced noise, confusing the smaller model. Reducing the parameter to 5 improved accuracy.

4 Experiments

4.1 Datasets

The **Criminal Law** subset of the **KMMLU** dataset [2] was used for this project.

Dataset Details

- **Source:** KMMLU (Korean Massive Multitask Language Understanding) - Criminal Law.
- **Total examples:** 200 Test samples. (The retrieval index was built using the Train split).

- **Train/Test split:** The standard KMMLU split provided was utilized.
- **Preprocessing steps:**
  - For retrieval indexing, "Question + Options" from the training set were concatenated.
  - The "Answer Label" (e.g., "Answer: A") was intentionally removed from the indexed text during optimization to prevent the model from biasedly copying the label from a retrieved (but different) question.

4.2 Metrics

Accuracy was used as the primary metric, which calculates the specific percentage of correctly predicted answers out of the total test samples.

4.3 Results

The Random Baseline was compared against the optimized AI Pipeline.

Method	Accuracy	Details
Random Baseline	24.5%	(49/200)
AI Pipeline (RAG + CoT)	**42.0%**	(21/50, subset)

Note: The AI Pipeline result is based on a verification subset of 50 samples.

Qualitative Analysis: The pipeline successfully retrieves relevant precedents. For example, when asked about exceptions to "Murder", the system retrieves cases discussing "Self-defense" or "Negligence" and correctly deduces that Negligence is not Murder. However, failures still occur on complex "twisted" logic questions where multiple laws interact, sometimes resulting in hallucinations of connections that do not exist, even with context.

5 Reflection and Limitations

Reflection

This project demonstrated that "System Engineering" is as important as the model itself. Initially, the pipeline performed poorly (20%) due to a concurrency bug and a poor prompt structure (Example placed after Question). Fixing these engineering issues and properly ordering the prompt (Rules $\rightarrow$ Example $\rightarrow$ Context $\rightarrow$ Question) effectively doubled the performance.

A major limitation is the dependency on the retrieval quality. If the Top-5 documents do not contain the exact answer, the model is forced to guess, often incorrectly. Furthermore, `gpt-4o-mini` sometimes struggles to ignore irrelevant "distractor" texts in the context. Future work could involve implementing a "Re-ranking" stage to filter the retrieved documents before sending them to the generator, or fine-tuning the embedding model on legal data to improve retrieval precision.

References

[1] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[2] HAERAE-HUB. Kmmlu: Korean massive multitask language understanding, 2024. URL https://huggingface.co/datasets/HAERAE-HUB/KMMLU.