Legal Knowledge and Information Systems
G. Sileno et al. (Eds.)

doi:10.3233/FAIA230978

Judgment Retrieval Made Easier Through Query Analysis

Tien-Hsuan WU [a], Ben KAO [a], and Michael MK CHEUNG [b]
[a] *Department of Computer Science, The University of Hong Kong*
[b] *Faculty of Law, The University of Hong Kong*

Abstract. The Hong Kong Legal Information Institute (HKLII) provides a repository of legal documents in Hong Kong and such as ordinances and historical court judgments. HKLII provides a search facility through which users retrieve relevant documents. We perform statistical analysis on HKLII access log over a 5-year period categorizing user search queries and discovering interesting user access patterns. Based on this study, we propose enhancing user experience through identifying the search intent of the user. We classify a user query into one of several types and customize the search behavior based on the query type to enhance user experience. Our study provides an example of leveraging log analysis in legal information system design.

Keywords. log analysis, legal information system, recommendation

1. Introduction

In common law jurisdictions, prior judgments (a.k.a. *precedents*) serve as a body of law. Legal practitioners therefore have to familiarize themselves with precedents. Due to the massive volume of historical judgments, machine-assisted search and retrieval of relevant judgments are an essential component in legal technology. The World Legal Information Institute (WorldLII) [1] is a consortium of fourteen legal information institutes, which provide legal professionals and the general public with free accesses to legal information, covering 123 jurisdictions and over 1,800 databases of legal information. Being a supporting member of WorldLII, The Hong Kong Legal Information Institute (HKLII) serves as the primary repository of legal documents, such as ordinances, regulations, and court judgments in Hong Kong. In this study we focus on judgment retrievals.

HKLII users can access historical court judgments on the website by *browsing* and *searching*. For browsing, HKLII provides a table of contents that organizes judgments by case names and dates of judgments. For searching, HKLII provides two query interfaces: a *simple keyword search* box and an *advanced search form* with which a user can input various fields (e.g., neutral citation, legislation). The simple keyword search box provides a fast and easy way for a user to retrieve the judgments containing some keywords, while the advanced search form allows a user to specify multiple criteria to narrow down the search scope. HKLII maintains a server log, which records user accesses to the website. From the log, we find that more than 95% of document accesses are done through searching rather than browsing. HKLII users, therefore, rely heavily on the search function to retrieve relevant documents. In addition, among all the searches, 73.7% are made through simple keyword search box. The simple keyword search box, despite its limitation that only a few keywords are allowed, is still heavily relied by the users. It is thus interesting to study user search patterns and investigate how user search experiences on HKLII can be enhanced. As an example, in one extreme case, a user searched for the phrase “umbrella contract” and then revised the query 11 times and accessed 79 judgments. To improve the user experience, we could make the system intelligently suggest some high-quality queries so that the user can revise the query with much less effort.

In this paper, we perform log analysis to identify the most popular categories of user queries. We also investigate what actions a user usually takes after submitting a query. Based on our analysis, we propose some improvements on HKLII’s search engine to enhance users’ search experience. Specifically, given a user query, the search engine would analyze the query type and take different actions according to the query type. We remark that our proposal can be applied to not only HKLII, but also to other legal information institutes (LIIs). We have investigated the search engine functionality of several LIIs, and have observed that they too encounter some similar issues.

2. Method

We analyze HKLII’s server log records that were collected from 2018 to 2022 (5 years). Each log record records an HTTP request submitted by a user (through a web browser) to HKLII’s web server. We extract the following information from each log record: (1) The requester’s **IP address**. (2) Request’s **timestamp**. (3) A **request line**, which is the HTTP request string. Typically, it contains the URL of the web page requested by the client. (4) An optional **referrer**, which indicates the web page the browser is displaying while the next HTTP request is sent. Through referrers, we can string together the actions taken by a user as a sequence of web page accesses.

We focus on studying court judgments search. Therefore, we filter the log and retain only those records that are associated with either *query submissions* or *judgment accesses*. In particular, the request line of a query submission log record contains the keywords mentioned in the user query, while that of a judgement access contains the *neutral citation* of the corresponding judgment. Next, we categorize the collected queries and recover user sessions from the log records. Each session reveals a sequence of page accesses done by a user in one sitting.

2.1. Query Classification

To understand what users usually look for in the judgment database, we classify each query into one of five categories:

Case Search. The query string is in the form of a neutral citation, a law report citation, a case number, or a case name. The HKLII search engine will return as search results (1) *target judgments*: those that match the search string, and (2) *citing judgments*: those that cite the targets. We compile a set of regular expression patterns to identify case searches by case numbers and the format of case names.

Legislation Search. The query string is an ordinance/regulation name or an ordinance/regulation number. Judgments that cite (deal with matters related to) the target ordinance are returned as search results. Similar to case searches, we also adopt a hybrid approach to identify legislation searches. Queries that search legislation by its number are identified by regular expression. For queries that search legislation by its name, we maintained a database that stores the names of the ordinances and regulations. Given a query, we first retrieve some best-matching legislation from the database using Elasticsearch. If any legislation contains all words in the query string, then the query is considered as a legislation search.

Entity Search. The query string contains names of persons or organizations. Judgments that concern the given entity would be returned as the results. Different from case searches (by case name) where both parties are given in the query string, the user may want to find all cases related to a person or organization. We identify entity search by applying the SpaCy Entity Recognizer [2] to recognize entity names. If the query string is labeled as `PERSON`, `NORP` (nationalities, religious or political groups), `ORG` (organizations), or `GPE` (geopolitical entities), then it is classified as an entity search.

Concept Search. The query string contains legal concept terms. When submitting a concept search, a user expects the system to retrieve judgments that deal with the concepts in the query. To identify concept searches, we build a taxonomy of legal concepts from ordinances. Concept terms are extracted from ordinances' section headings, their interpretation sections (which define the terms used in the ordinance) and glossary [3].

Others. The search string cannot be classified as any of the four types.

2.2. Session Construction

We construct *user sessions* from the log records. Specifically, a session S is a sequence of log records that are associated with a user's accesses to HKLII in one sitting. We denote a session by $S = [q \mid a_1, ..., a_n]$, which models a user submitting a query q to HKLII on its homepage followed by a number of *actions* $a_1, ..., a_n$ taken subsequently by the same user. We call q the *session head* and $< a_1, ..., a_n >$ the *session body*. The number of actions n is the *length* of the session. After a user posts a query q, the user can do one of two things as his actions: he/she can either access a judgment or a legislation, or modify his/her query and submit the revised query. A session ends when the user returns to HKLII homepage or leave HKLII. In addition, if a user idles 6 hours after an action is made, then we consider the user ends the session and starts a new one. To exclude sessions that might be initiated by web crawlers, we remove sessions of lengths 50 or longer. We also remove sessions with empty bodies as those do not contribute information to our analysis. In total, we recovered about 1.46 million sessions from the 5-year server log records.

3. Log Analysis and the Design of Intelligent Search Engine

In this section, we present the results of the log analysis and discuss how the HKLII's search engine has been improved. The search engine on HKLII is powered by Elasticsearch, which provides a number of options for us to adjust its search behavior. Given

Session Category	(a) Session Count	(b) Average Session Length	(c) Long (10+) Session Ratio	(d) Multi-query Session Ratio
Case Search	386,989 (25.9%)	2.19	3.1%	27.1%
Concept Search	396,832 (26.6%)	5.16	17.1%	47.8%
Entity Search	357,410 (24.0%)	3.21	7.4%	36.5%
Legislation Search	67,441 (4.5%)	6.16	21.1%	35.5%
Others	283,029 (19.0%)	4.76	14.7%	48.9%
All	1,491,701	3.79	10.4%	40.4%

Table 1. Session statistics (2018-2022)

an input query, the query analyzer classifies each query into the five categories in Section 2.1 and chooses the most suitable search behavior parameters according to the query type.

We classify each session into one of 5 categories (*Case, Legislation, Entity, Concept*, and *Others*) according to the query of the session head. Table 1 shows some statistics. Column (a) shows the number of sessions for each category and the percentage (in parentheses) w.r.t. the total query count. We see that case/entity/concept searches are quite evenly split — each accounts for roughly 25% of all queries. Legislation search (4.5%) is much less frequent but has the highest average session length. Column (b) shows the average session lengths. We see that case search sessions (2.19) are generally much shorter than others. On average, a user performs just about 2 actions in a case search compared with, for example, concept search (5.16) for which a user performs nearly 5 actions on average. We further define a *long session* to be one whose length is 10 or above; and a *multi-query session* as one that contains at least one (modified) query in its session body. Long and/or multi-query sessions are interesting because they represent *difficult searches* — those that require enquirers to read many judgments and/or make adjustments to their query keywords. Columns (c) and (d) show the percentages of long and multi-query sessions for each query category, respectively. From the table, we see that a significant fraction (17.1%) of concept searches are long and that about half of the time (47.8%) a concept search enquirer does not get the search keywords right and has to refine and resubmit his query. Based on the quantitative analysis result, we further inspect the log and make the following observations.

3.1. Case Search

Case searches are very short (2.19) and their queries are the least likely to be refined (27.1%). This is because case search is very specific—a user provides an explicit identifier to retrieve a specific judgment. There are two purposes when a user would like to issue a case search. First, with such a specific query, it can be inferred that the user already knows the judgment to read. The search box would simply be a tool for the user to locate the judgment without the need to browse the table of contents. Secondly, a user may want to look up an influential case that is cited by many judgments. This would make the session length of case searches larger as the user often reads multiple judgments to understand how the search target is cited by the judges in different scenarios.

Based on our observation, we further investigate the search engine and propose the following improvement for case searches. Since the default behavior of Elasticsearch is full-text search over the judgments, the top-ranked judgments could be the ones that mention the search target multiple times. These citing judgments would therefore considered to better match the input query. When a user simply intends to access the judgment by inputting the case identifiers to the search box, he/she still needs to scan through the returned judgments in order to locate the judgment. To improve the user experience, we propose *title boosting* for case searches. If the query classifier identifies that the query is a case search, then it also passes a boosting criteria to the search engine such that the judgment would be ranked higher if the query matches the title of the judgment. With this design, users can save some effort when using the search box to directly access the judgment, while others looking for citing judgments can still be satisfied.

3.2. Concept Search

Concept searches are generally much longer (17.1% long sessions) and the queries often have to be refined (47.8% multi-query sessions). On further investigation, we observe that many enquirers of concept search fail to choose the right combinations of keywords to express their search intents. As a result, an enquirer may have to read irrelevant judgments in the search results and often have to refine the query with alternative keywords.

We propose to apply data mining techniques to perform intelligent *query recommendation*. The idea is to have the machine analyze the historical search log to learn about query similarity and popularity and build a keyword recommendation database. From the search log, we collected all the conjunctive queries that led to at least one judgment access and tallied the co-occurrence counts for each pair of keywords. For example, for the query "*partnership* and *accountant* and *trust*", one count will be recorded for the three pairs: (*partnership*, *accountant*), (*accountant*, *trust*), and (*partnership*, *trust*). We excluded queries that user issued without accessing judgments as a user would less likely to be satisfied with such a combination of keywords. To avoid associating a legal concept with an entity, we further removed keyword combinations that contain person or organization names. Given a user concept search query, the system would deduce the most popular queries that co-occur with the user input and recommend them to the user.

3.3. Entity Search

We investigate entity searches that have long and/or multi-query sessions. Typically, an enquirer inputs an entity name (of a person or an organization) to retrieve judgments that involve the entity. As the transliteration of the Chinese name to English is not canonicalized (e.g., "Fong" and "Fung"), there could be typos in the name, resulting in multiple queries in the session. This would also happen when different names have a similar pronunciation (e.g., "Carlton" and "Carton"). To resolve this issue, our proposal is to employ fuzzy matching for entity searches. In fuzzy searches, we specify a fuzziness value ε such that two words are consider a match if their Levenshtein edit distance is no greater than ε. Given a query comprising of n words $[w_1^q, w_2^q, \ldots, w_n^q]$, a judgment is considered a match if there exists a sequence of n words in the judgment $[w_1^j, w_2^j, \ldots, w_n^j]$ such that $\mathit{Levenshtein}(w_i^q, w_i^j) \leq \varepsilon \quad \forall i \in 1, \ldots, n$. In this design, fuzzy matching are applied at the word-level: the fuzziness ε is allowed for each word. Therefore, we set ε to 1, a small value. The fuzzy matching tolerates some spelling mistakes so that the enquirer is not required to find out the correct spelling of an entity before initiating a search on the judgment database system.

In addition to the fuzzy matching, we also adopt title boosting as we have discussed in case search. When an entity name is entered as the search string, it is more likely that the user intends to find the judgments that the entity is one of the two parties. Still, in some scenario the user would be searching for judgments that cite the case in which the query entity is involved. When displaying the results to the user, we show the judgments having the query string in the title, followed by judgments with the search string in the context. We also display judgments that exact match the query string before judgments matching the query string with fuzziness.

3.4. Legislation Search

A legislation search usually specifies an ordinance as the query with the names or the ordinance number. The user either intends to read the given ordinance, or to retrieve judgments that cite the given legislation. From the analysis of the query log, we see that legislation search has a high average session length (6.16). We observe from the query log that users often use the search box as a way to directly accessing the ordinances, instead of clicking the links of the ordinances in the table of contents. In the database, an ordinance is split into sections and stored as separate pages, and each is displayed as an item in the search results. This design is to facilitate the retrieval of ordinances in segments so the user can search for a particular section of an ordinance. However, users will need to click on multiple items in the search results to read multiple sections, which is not desirable for those who intend to read a significant part of the ordinance. Considering this use case, it would be convenient for the user if the system displays a link that directs the user to the full content of the ordinance, so that the user can choose to read a specific section in the search result, or to read multiple sections in a single page.

4. Conclusion

In this paper, we performed log analysis on HKLII to discover user access patterns. From the log records, we recovered more than one million query sessions. We investigated the long and multi-query sessions, which represent the most difficult searches. Based on our analysis, we improved the user search experience by classify queries and designing different approaches to handle different types of queries. Our design has been implemented on HKLII, and the methods can apply to other legal information systems such as the other members of WorldLII.

References

[1] WorldLII. World Legal Information Institute; 2020. Available from: www.worldlii.org/.

[2] Honnibal M, Montani I, Van Landeghem S, Boyd A. spaCy: Industrial-strength Natural Language Processing in Python; 2020.

[3] English – Chinese Glossary of Legal Terms. Department of Justice; 2020. Available from: https://www.elegislation.gov.hk/glossary/en.