Legal Knowledge and Information Systems
E. Francesconi et al. (Eds.)

doi:10.3233/FAIA220476

Judgment Tagging and Recommendation Using Pre-Trained Language Models and Legal Taxonomy

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

Abstract. We study the problem of machine comprehension of court judgments and generation of descriptive tags for judgments. Our approach makes use of a legal taxonomy $\mathcal{D}$, which serves as a dictionary of canonicalized legal concepts. Given a court judgment J, our method identifies the key contents of J and then applies Word2Vec and BERT-based models to select a short list T_J of terms/phrases from the taxonomy $\mathcal{D}$ as descriptive *tags* of J. The tag set T_J suggests concepts that are relevant to or associative with J and provides a simple mechanism for readers of J to compose *associative queries* for effective judgment recommendation. Our prototype system implemented on the Hong Kong Legal Information Institute (HKLII) platform shows that our method provides a highly effective tool that assists users in exploring a judgment corpus and in obtaining relevant judgment recommendation.

Keywords. judgment recommendation, judgment tagging, keyword extraction

1. Introduction

In common law, prior judgments (a.k.a. *precedents*) serve as the body of law following which courts make decisions with similar judicial reasoning. This principle (called *stare decisis*) makes it important that legal professionals be able to effectively find relevant judgments as references. There are a number of online platforms such as the World Legal Information Institutes (WorldLII) that provide online accesses to historical court judgments. These platforms generally provide tools for users to retrieve judgments with keyword search. In practice, however, the task of finding reference judgments for legal research is often *concept-based* instead of *instance specific*. For example, one may want to find previous cases of "*teenagers trafficking illegal drugs*" instead of specifically a case in which "*an 18-year-old was found carrying 5.2g of cocaine*" as the latter is so specific that a search will unlikely net any reference judgments. Traditional keyword-search tools do not handle concept-based search well as they require stringent string matching against the textual contents of judgments, which by nature is specific to each judgment. Another characteristic of judgment search is that the exercise is often *exploratory*, with the search goal not perfectly identified at the start but incrementally refined as one progresses. A lawyer who has found a judgment J that partially matches his/her search intent may want to expand his collection with other judgments that are associative with certain

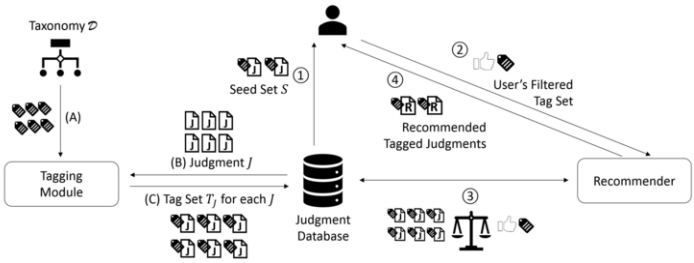

Figure 1. CAJUR architecture

details of J. For example, a lawyer researching for personal injury cases may have retrieved a judgment on *metatarsal bone* fracture because that is an injury that the lawyer's client has sustained. However, the client's injury may also involve *tibialis anterior* (a muscle attached to the metatarsal) tear. In this case, "tibialis anterior" is a concept that is *associative with* "metatarsal bone". Linkages from one judgment (e.g., on metatarsal bone) to others associative judgments (e.g., on tibialis anterior) would greatly facilitate the lawyer's reference judgment search.

In this paper we study the problem of **concept-based associative judgment recommendation**. Our goal is to design a system that assists legal professionals in effectively finding reference judgments during legal research. We assume that a user starts with a few judgments that satisfy his/her initial search goal. We call this initial set of judgments the *seed set* $\mathcal{S}$. A user can obtain $\mathcal{S}$ by any means, such as browsing or keyword searches. The problem is to recommend other judgments J's to the user with the requirement that the concepts covered in J's are either the same as or associative with those of the judgments in the seed set. Our proposed approach of concept-based associative judgment recommendation can help users in better expressing their search intents and exploring the judgment database resulting in more effective and efficient reference judgment search.

2. Method

In this section we describe our method CAJUR, which stands for *Concept-based Associative Judgment Recommendation*. CAJUR performs judgment tagging for effective recommendation. Figure 1 illustrates CAJUR's design. We start with a judgment database. CAJUR uses a legal taxonomy $\mathcal{D}$ as a dictionary of standard tag terms/phrases (A). It employs a tagging module that selects phrases from the taxonomy to create a tag set T_J for each judgment J in the database (B&C). Let $\mathcal{S}$ (①) be a seed set of judgments that are relevant to a user's search intent. Each judgment $J' \in \mathcal{S}$ is also given a tag set $T_{J'}$ by the tagging module. The user can compose a query (②) by including or excluding some tags in the tag sets $T_{J'}$'s. The recommender module (③) will then compare the query against the tag set T_J of each judgment J in the database to determine whether J should be recommended to the user (④). Next, we give technical details of the tagging module and the recommender module.

2.1. Tagging Module

Let $\mathcal{D} = \{p_1, p_2, ...p_{|\mathcal{D}|}\}$ be a legal taxonomy that consists of a collection of phrases p_i's. Given a judgment J, the tagging task is to determine a subset $T_J \subset \mathcal{D}$ such that each phrase

$p \in T_J$ is *relevant to* or *associative with* the content of J. Specifically, the tag set should consist of two parts $T_J = T_J^r \cup T_J^a$. The set T_J^r is a collection of *relevant tags* each of which describes a concept that is mentioned in the judgment. T_J^r helps the user grasp the major elements of J and to obtain recommendation of other judgments that are similar to J. On the other hand, the set T_J^a gives a collection of *associative tags*. These tags are not explicitly mentioned in the judgments but they describe concepts that are associative with those mentioned in J. The set T_J^a helps the user expand his/her search by considering related concepts that are not explicitly included in J. The tagging process consists of two steps, namely, (1) *sentence selection* and (2) *tag prediction*. To facilitate our discussion, we will illustrate our method using *personal injury compensation (PI) cases in Hong Kong* as an example.

2.1.1. Sentence Selection

Judgments are generally long and complex. To abstract a judgment, which may contain thousands of words, to a handful of concept tags requires techniques that can identify the key aspects and elements of a judgment. The first step of the tagging module is to extract a set of key sentences $KS_J = \{s_1, ..., s_{|KS_J|}\}$ from a judgment J that cover the major aspects of a court case. For example, for PI cases, key aspects include *plaintiff background*, *injury*, *loss*, *treatment* and *compensation*. We employ the technique given in [1] to identify key sentences for each given aspect in a judgment. Due to space limitation, readers are referred to [1] for details. In the following discussion, we focus on *injury* and *loss* aspects for illustrative purpose.

2.1.2. Tag Prediction

Given the taxonomy $\mathcal{D}$ and the key sentences KS_J, the next step is to determine the semantic correlation between each phrase $p \in \mathcal{D}$ and each sentence $s \in KS_J$. Those p's that are of high semantic correlation with the s's are collected in the tag set T_J. Recall that T_J consists of two subsets T_J^r and T_J^a. The tagging module thus uses two taggers, namely, a Word2Vec tagger and a BERT tagger to perform respective semantic analysis.

[Word2Vec tagger] The objective of the Word2Vec tagger is to generate the tag set T_J^r, which represents concepts that are mentioned in the judgment J. For each sentence $s \in KS_J$ that is extracted under a certain aspect A of the case, we consider the relevant section $\mathcal{D}_A$ of the taxonomy $\mathcal{D}$. For example, for the aspect *injury*, we consider a subset $\mathcal{D}_{\text{injury}} \subset \mathcal{D}$, which consists of 382 phrases that express various kinds of injuries; for the aspect *loss*, we consider another subset $\mathcal{D}_{\text{loss}}$, which consists of 41 phrases. Given a sentence s, we obtain all the noun phrases (using TEXTBLOB) contained in it. For each such noun phrase np, we compute the cosine similarity between the Word2Vec embeddings of np and each phrase $p \in \mathcal{D}_A$. If a phrase $p^* \in \mathcal{D}_A$ gives the highest score and if the score exceeds a threshold τ, we consider the phrase p^* and the sentence s semantically correlated. We thus put p^* in T_J^r. The threshold τ is determined empirically; In our prototype, we set $\tau = 0.6$. Note that the tags generated by the Word2Vec tagger express concepts that are directly relevant to the extracted sentences. Next, we describe a BERT tagger that finds tags that are associative with the sentence's contents. For example, the tag "loss of leisure" should be obtained if the judgment mentions "can't enjoy life".

[BERT tagger] We use the BERT Next Sentence Prediction (BERT-NSP) architecture to determine the semantic correlation between a phrase p and a sentence s, particularly

for phrases that express concept that are not explicitly mentioned in s. The BERT-NSP architecture has been used in sentence pair classification tasks. The model takes two text sequences as input and outputs a classification result. In our study, we train a BERT-NSP model that classifies if p and s are semantically correlated. Specifically, given a (p, s) pair, the classifier outputs a confidence score. If the score exceeds a threshold σ and if p is not already collected in T_J^r by the Word2Vec tagger, then p is considered an associative tag and is put in T_J^a. In our prototype, we set $\sigma = 0.8$. Furthermore, as there are potentially many associative concepts, we retain only the top-5 tags based on their confidence scores. To train the BERT tagger, we need a training set that consists of positive and negative samples. We collect the sentences and the phrases extracted by the Word2Vec tagger as the positive samples. For each sentence, we further sample 5 other phrases in $\mathcal{D}$ and include them as the negative samples. We trained six sub-models using different hyper-parameters, and our final BERT tagger outputs a phrase p for a sentence s if at least two of the sub-models vote for p.

2.2. Recommender Module

Judgment recommendation is made based on a user's search intent. For that, a user maintains a *focal tag set* F and an *exclusion tag set* E. Intuitively, we find judgments that mention the concepts expressed by the tags in F but not those in E. As a user explores the judgment database and reads a judgment J, the user can refine the F and E sets by including or excluding the tags given in T_J. With the F and E sets, the recommender module retrieves a set of candidate judgments R from the judgment database DB. R is given by: $R = \{X \in DB \,|\, F \subseteq T_X \text{ and } E \cap T_X = \varnothing\}$. Each judgment X in R is then ranked based on the following scores H_1 to H_3:

(Relevant tag count): $H_1 = |F \cap T_X^r|$. Recall that the relevant tags T_X^r (generated by the Word2Vec tagger) are those that express concepts directly mentioned in the judgment X. The higher the H_1 score, the more directly relevant is X to the user's search focus.

(BERT tagger votes): H_2 = total number of votes given by the 6 BERT models to the tags in F when the BERT models predict tags for judgment X. The higher this vote count, the more confident we are that X is relevant or is related to the concepts expressed in F.

(Average tag similarity): $H_3 = \text{AVG}(cs(t_a,t_b) \,|\, t_a \in F \text{ and } t_b \in T_X)$, where $cs(t_a,t_b)$ is the cosine similarity of the Word2Vec embedding vectors of tags t_a and t_b. H_3 measures the average semantic closeness of the focal tags and those of judgment X. Judgments in R are ranked first by their H_1 scores, with H_2 and H_3 serve as level 1 and level 2 tie-breakers.

3. Demonstration and Evaluation

We implemented a prototype of CAJUR on the HKLII platform. In this section we briefly describe the interface. Also, we evaluate the effectiveness of the tagging module and the recommendation module through experiments.

Figure 2 shows a screenshot of the interface where a PI judgment J is displayed in the middle. The tag set T_J, which consists of 9 tags, is shown in the right pane. Among them, the relevant tags T_J^r are displayed in darker blue while the associative tags T_J^a are in lighter blue. The taxonomy $\mathcal{D}$ from which tags are obtained provides a hierarchy of phrases with related concepts grouped under a subtree. Tags in T_J that share common

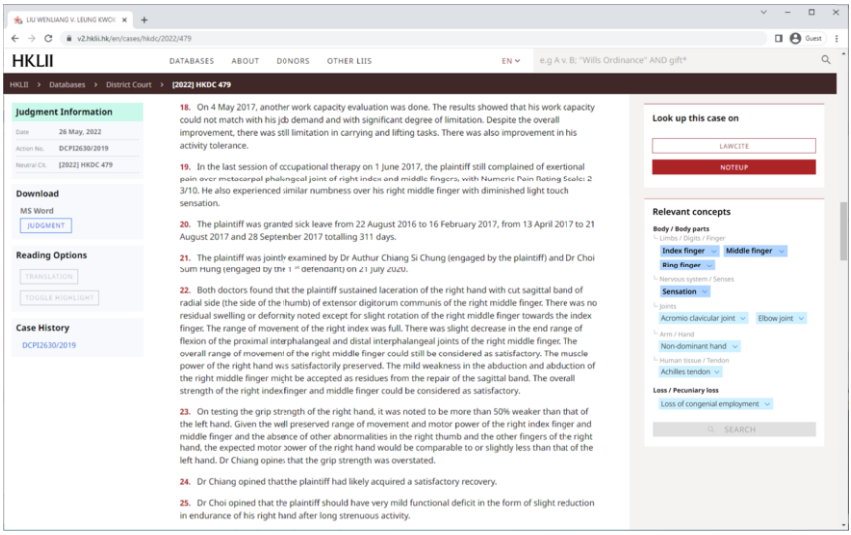

Figure 2. Interface showing a judgment and its tags

Table 1. Tag evaluation

	PS	PSLegal	CAJUR
Relevant	76 (30.8%)	75 (30.4%)	70 (28.4%)
Associative	4 (1.6%)	5 (2.0%)	49 (19.8%)
Others	167 (67.6%)	167 (67.6%)	128 (51.8%)

Table 2. Recommendation quality

	Avg. Score
Document Similarity	0.48
Tag-Doc Similarity	0.26
CAJUR	1.16

path prefix in the hierarchy are displayed as a group with the path prefix shown to provide contexts. For example, the tags *Index finger*, *Middle finger*, and *Ring finger* are child nodes of the branch *Limbs* → *Digits* → *Finger*. A user can put a tag into his focal tag set F or his exclusion tag set E by either *including* or *excluding* a tag, respectively. The user can then click the search bar and the system in response will display a short list of recommended judgments.

We conduct experiments to evaluate the effectiveness of CAJUR's tagging module and recommendation module. For tag quality, we compare CAJUR against two methods, namely, PS and PSLegal [2], which have been shown to be effective methods in identifying key phrases in legal documents. We sample 25 PI judgments from Hong Kong courts. For each judgment J, CAJUR generates a tag set T_J. We then apply PS and PSLegal to extract $|T_J|$ key phrases from J for each method. All phrases given by CAJUR, PS, and PSLegal are collected and shuffled before they are presented to human evaluators in a single list. We employ six human evaluators who are asked to classify each tag (phrase) with the following descriptors: **Relevant tag**: The tag helps the user grasp an important element/concept mentioned in the judgment; It is conceivable that the user would select the tag in expressing a search intent for similar judgments. **Associative tag**: The tag does not express elements/concepts that are directly mentioned in the judgment but is related to or associative with them; It is conceivable that the user would select the tag in extending a search intent in search of other related judgments. **Others**: It is unlikely that the user will select the tag in expressing a search intent.

Table 1 shows the total number of each tag category given by each method as evaluated by the human evaluators. We see that CAJUR generates significantly more associa-

tive tags than PS and PSLegal. CAJUR is therefore much more effective in suggesting tags to users in formulating *associative queries*. This will greatly help users in exploring the judgment database. As an illustrative example, there are a number of PI judgments on HKLII that contain the phrases "upper back" and "neck". For these judgments, CAJUR generates the tag "trapezius muscle", which is a large muscle piece that connects the neck and the upper back. This associative tag helps connect the back-and-neck judgments to those that mention trapezius muscle injuries. The ability of CAJUR in generating associative tags thus help users identify related judgments which would otherwise be missed if only straightforward keyword matching were done.

Finally, we evaluate the accuracy of CAJUR's recommendation module. For each of the 25 sampled judgments used in the tagging experiment, we select 1-2 tags into a focal tag set F. We then recommend two judgments using CAJUR and the following two baseline methods. **Document Similarity**: We embed the sampled judgment and all judgments in the corpus using Doc2Vec [3]. The two judgments that are most similar to the selected sampled judgment are selected. **Tag-Doc Similarity**: We embed the focal tags and all judgments in the corpus using Doc2Vec, and select two judgments that are most similar to the focal tags. We collect the recommended judgments from all approaches and shuffle them before presenting them to a human evaluator for evaluation. The evaluator was asked to decide whether each recommended judgment is *relevant to the focus* (2 marks), *somewhat relevant* (1 mark), or *irrelevant* (0 marks).

Table 2 shows the results. We see that CAJUR's recommendation module is significantly more effective compared with the baselines. We observe that Tag-Doc Similarity does not perform as well as Document Similarity. The reason is that the Doc2Vec embedding vectors of some words (i.e., tags) can be very different from those of entire documents (i.e., judgments) as the word distributions of the two are quite different. This shows that recommendation approach with tagging needs to be thoughtfully designed. CAJUR generates more precise recommendations by fully using the tags assigned to the judgments in the corpus as the recommendation criteria.

4. Conclusion

In this paper we study the problem of tagging and recommending legal judgments. We propose CAJUR, which achieves the tasks by using a taxonomy and the BERT model. CAJUR generates relevant tags that express concepts mentioned in judgments as well as associative tags that provide users hints on how their search could be extended with associative queries. We present a user interface that displays tags and facilitates a user to specify a search focus. The evaluation results show that CAJUR can generate high quality relevant and associative tags, as well as recommend judgments that are much more relevant to the user search focus.

References

[1] Wu TH, Kao B, Chan F, Cheung A, Cheung M, Yuan G, et al. Semantic Search and Summarization of Judgments Using Topic Modeling. In: Legal Knowledge and Information Systems. IOS Press; 2021. .

[2] Mandal A, Ghosh K, Pal A, Ghosh S. Automatic catchphrase identification from legal court case documents. In: CIKM; 2017. p. 2187-90.

[3] Le Q, Mikolov T. Distributed representations of sentences and documents. In: International conference on machine learning. PMLR; 2014. p. 1188-96.