Learning to Rank Sentences for Explaining Statutory Terms

Jaromir Savelka [a,1], and Kevin D. Ashley [b,2]
[a] *School of Computer Science, Carnegie Mellon University*
[b] *School of Law, Intelligent Systems Program, University of Pittsburgh*

Abstract. We explore using classical feature engineering-based learning-to-rank approaches (LTR) to discover sentences for explaining the meaning of statutory terms. We compiled a list of 129 descriptive features that model retrieved sentences, their relationships to statutory terms, and their statutory provisions of origin. Using a statutory interpretation data set (26,959 sentences) we showed how the proposed feature set could be utilized in learning-to-rank settings with reasonable effectiveness. We showed that off-the-shelf machine learning algorithms perform significantly better than BM25 baselines (NDCG@100 of 0.77 vs 0.68).

Keywords. Information retrieval, Learning-to-Rank, Statutory Law, Interpretation

1. Introduction

Statutory provisions express legal rules. Understanding statutes is difficult because the abstract rules they express must account for diverse situations, even those not yet encountered. Interpretation may help to deal with doubts about what a provision means [12]. It involves investigating how a term has been referred to, explained, or applied in the past. A lawyer constructs arguments in support of or against particular interpretations. For example, consider the two emphasized phrases from the following (abridged) example provision (29 U.S. Code § 203):

> "*Enterprise*" means the related activities performed [. . .] for a *common business purpose* [. . .].

The meaning of *common business purpose* may determine if two separate restaurants with a single owner constitute an *"enterprise"* within the provision's meaning.

Searching through a database of statutes, court decisions, or law review articles, one may stumble upon sentences such as these:

(i) [. . .] a joint profit motive is insufficient to support a finding of *common business purpose*.
(ii) The problems then are whether we have related activities and a *common business purpose*.
(iii) The third test is "*common business purpose*."

[1] jsavelka@andrew.cmu.edu
[2] ashley@pitt.edu

Proceedings of the 2020 Workshop on Automated Semantic Analysis of Information in Legal Text (ASAIL)

Sentence (i) is useful for the interpretation of the term because it elaborates on a condition which is not sufficient to meet the requirements of the term. Sentences (ii) and (iii) do not appear to be useful. Reviewing the list of results manually is labor intensive and may involve hundreds or thousands of documents.

This work follows on our analysis of the problem in [19], the initial proof of concept system [18], and the attempt to tackle the task via traditional IR approaches [15]. Here we investigate if a system can respond to a user-specified statutory phrase with useful sentences for understanding its meaning. We treat the task as a learning-to-rank (LTR) problem using a data set of 26,959 sentences. We compiled a list of descriptive features and trained and evaluated a number of ML algorithms. We have shown that a sentence classification system can learn to rank sentences according to their utility.

2. Related Work

Our legal information retrieval (IR) application uses argument mining to improve performance. The authors of [1] advocate legal argument retrieval (AR) systems as the next stage in legal IR since lawyers are primarily interested in retrieving arguments. Subsequent work demonstrated AR systems for the domains of vaccine injury compensation and veterans disability compensation (See, e.g., [7,2,22]). We, however, do not employ granular domain-specific argument models such as sentence type systems and rule trees.

Walter [23] focuses on extracting definitions from non-fact-reporting sections of German court decisions. Like us, he extracts definitions directly from decision texts, not via intermediary human-crafted representations. He works with only three definition types (Core, Addition, Definiendum) and, like us, does not employ domain-specific argumentation models, making the work domain agnostic.

The effects of using query or document contexts in legal IR have been explored extensively in AI & Law literature. The authors of [20] experiment with query expansion using lexical ontologies. The authors of [14] employed similar techniques to solve the problems of word synonymy and ambivalence. In [9] retrieval context is utilized to obtain relevance feedback from users. We show how query expansion (using the source provision) and a sentence's context improves the performance of the retrieval system.

3. Data Set

We employed the Caselaw access project,[3] including all official, book-published U.S. cases from federal, state, and some territorial courts [24]. We ingested the data set of more than 6.7 million cases into an Elasticsearch instance.[4] To analyze textual fields we

[3] A small portion of the data set is available at `case.law`. The complete data set could be obtained upon entering into research agreement with LexisNexis.

[4] `https://www.elastic.co/`

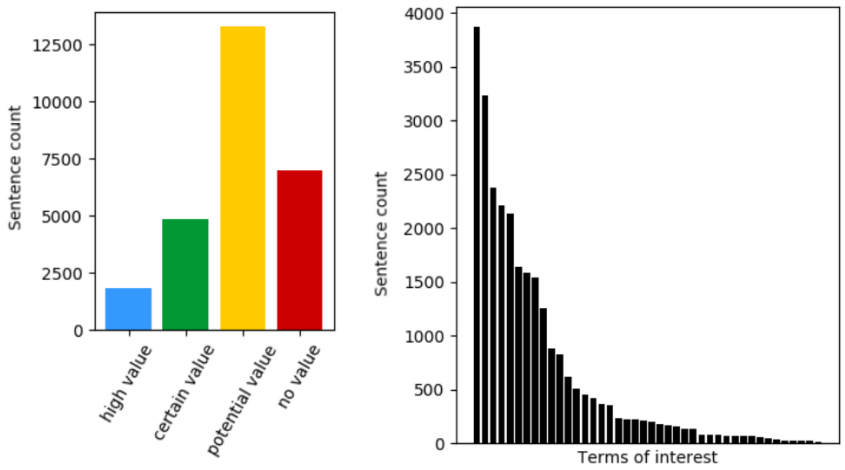

Figure 1. Overall distribution of labels (left) and number of sentences associated with each query (right).

used the LemmaGen Analysis plugin[5], a wrapper around a Java implementation[6] of the LemmaGen project.[7]

We indexed the documents at multiple levels of granularity, including at the level of full cases and as segmented into the head matter and individual opinions (e.g., majority opinion, dissent, concurrence).[8] Using the U.S. case law sentence segmenter [17] we divided each case into individual sentences. We also treated line-breaks between sentences as indicating new paragraphs.

We queried the system for all sentences mentioning any of 42 selected statutory terms (e.g., "audiovisual work," "electronic signature"). The terms were selected from provisions of the U. S. Code (the official collection of federal statutes)[9] covering different regulatory areas. All were vague terms. Due to the high cost of labeling, we used only these terms. The resulting data set comprised 26,959 sentences.

Eleven law students classified the sentences by utility for explaining statutory terms:

1. **High value** – Sentence elaborates the term's meaning.
2. **Certain value** – Sentence provides a basis for concluding what the term means.
3. **Potential value** – Sentence provides extra info beyond that in the provision.
4. **No value** – Sentence provides no additional information.[10]

Figure 1 shows the overall distribution of the labels and the number of sentences associated with each query. The Figure shows that the less valuable categories, 'no value'

[5] `https://github.com/vhyza/elasticsearch-analysis-lemmagen`

[6] `https://github.com/hlavki/jlemmagen`

[7] `http://lemmatise.ijs.si`

[8] This segmentation was performed by the Caselaw access project using a combination of human labor and automatic tools as per an email from info@case.law on 2019-01-07.

[9] `https://www.law.cornell.edu/uscode/text/`

[10] The annotation guide is at `https://github.com/jsavelka/statutory_interpretation/blob/master/annotation_guidelines_v2.pdf`.

and 'potential value', are dominant. These values are more numerous for all of the terms with larger numbers of queries and most terms with small numbers of queries. Some terms contain many more valuable sentences (e.g., "audiovisual work" or "switchblade knife") than others (e.g., "essential step" or "hazardous liquid").

4. The Features

In classical LTR one usually starts by identifying a list of features indicating relevance of retrieved documents. For example, [13] provides a hand-crafted list of features encoding the relevance-oriented relationship between the query and a document in the context of web search. These include generally applicable features such as TF-IDF scores of various parts of the document, lengths of those parts, or their language model matches to the query. Features such as inlink and outlink counts or PageRank scores are specific to the web search domain. Here we craft a list of 129 features that are relevant for retrieving useful sentences for arguing about the meaning of statutory terms.

Most of the features (59) are based on individual text units (e.g., the retrieved sentences, the query, the containing paragraphs). These include units' low-level characteristics like word counts and syntactic attributes, such as the unit root's POS type, as well as more sophisticated traits including the item's membership in specific functional parts of the decision (e.g., factual background, analysis as described in [16]). Additional 46 features focus on the interaction between items and the query. These features comprise different matching scores (e.g., BM25), the relationship of the query and quotes within the item, and syntactic attributes related to the position of the query within the unit (e.g., subtree size). We defined 19 features to model the relationship between the source provision and other units. The features are focused on the topical match between the source provision and the item or the novelty of the individual unit with respect to the source provision. Finally, five features model aspects of the retrieved results list (e.g., number of retrieved sentences, sentence/case ratio).

5. Experiments

The simplest LTR approach to ranking is to represent each document as a single feature vector. Given the existence of ground-truth labels, one can treat the problem as a standard classification or regression task. The goal is to learn a function that takes the feature vector of a document as input and predicts its relevance degree. Based on such a function, one can sort the documents into a ranked list. The ranking may be modeled as regression, classification, or ordinal regression/classification [11]. Almost any standard regression or classification algorithm can be used and many have been tried: Linear regression, Support Vector Machines, Logistic regression, Decision tree ensembles, or Neural networks.[11] In this section we report the results of the experiments on applying several of the regression

[11] The full list of algorithms and references is reported in [11].

and classification methods.[12] These take the features described in Section 4 as inputs and learn to predict sentences' value.

We first retrieve all the sentences that contain the term of interest. The individual sentences are then represented as features described in Section 4. Using different off-the-shelf ML algorithms, sentences are ranked from those predicted to be most valuable to least valuable. As baselines for reference we report the performance of the BM25, BM25-c, and Random methods. The two variants of BM25 are very close to what is typically used in many IR systems. Hence, they are effective baselines often not easily outperformed. BM25-c is a (linear) combination of the plain BM25 and another BM25 measure applied to the whole text of a case (i.e., sentence's context). Each method is evaluated via cross-validation in each step of which the algorithm uses one of the six folds as a test set. The remaining folds are used for training and optimization of hyperparameters.

Since the notion of relevance in this work is non-binary, we use normalized discounted cumulative gain (*NDCG*) to evaluate the performance of different approaches. We chose to evaluate the rankings at $k = 100$, which means that the tuples produced by the ranking algorithms are truncated to the respective lengths. For each query q_j, the *NDCG* at k is then computed as:

$$NDCG(S_j,k) = \frac{1}{Z_{jk}} \sum_{i=1}^{k} \frac{rel(s_i)}{log_2(i+1)}$$

The function $rel(s_i)$ takes a sentence as an input and outputs its value in a numerical form (0 for 'no,' 1 for 'potential,' 2 for 'certain,' and 3 for 'high value' sentence). Z_{jk} is a normalizing quantity which is equal to $NDCG(S_j,k)$ where S_j is the ideal ranking. The final evaluation metric is simply computed as a macro average over the set of queries. The procedure yields the NDCG@100 for the following groupings of queries based on the number of retrieved sentences and the number of sentences with higher value: small sparse, small dense, large sparse, large dense, and overall. We use the Friedman test [6] in combination with the Holm's step down method [8] as a post hoc test on the overall NDCG@100 metric to assess statistical significance.

First, we employ regression-based algorithms. In this setup the relevance of a document is regarded as a continuous variable. We evaluate the linear regression model with L2 regularization. The prediction $\hat{y}_i$ is understood as the respective sentence's score, which determines the order of the sentence in the ranking. We also tried an AdaBoost regressor [4] with decision tree regressor as the base estimator [3].

A second class of models we analyze are classification-based algorithms. These regard the relevance of a document as a categorical variable. In prediction we obtain the per-class probability vector ($\vec{p}_i$) for sentence s_i:

$$(p(s_i = \text{'no value'}), p(s_i = \text{'potential value'}), p(s_i = \text{'certain value'}), p(s_i = \text{'high value'}))$$

For the sentence's score we compute an inner product between $\vec{p}$ and value weight vector:

$$\vec{p}(0,1,2,3)^T$$

[12] For all the methods used in this paper we work with the implementations from the scikit-learn library which is to be found at `scikit-learn.org`.

This considers not only the predicted class but also the confidence in the prediction.

We use a logistic regression model with L2 regularization. We also evaluate support vector machines (SVM), another linear model. An SVM classifier constructs a hyperplane in a high dimensional space, which is used to separate the classes from each other. Of the non-linear models, we work with a random forest model. Random forest, an ensemble classifier, fits a number of decision trees on sub-samples of the data set. It uses averaging to improve the predictive accuracy and control over-fitting. Finally, we evaluated a multi-layer perceptron classifier. Unlike logistic regression, it includes one or more hidden layers between the input and output layers.

We use the same models in a setup that accounts for the labels' ordinal nature [5]. The task is transformed from a single k-class classification problem to $k-1$ binary classification problems. Three sets of labels use these transformation functions:

$$f_1(s_i) = \begin{cases} 0 & \text{if } s_i \text{ has no value} \\ 1 & \text{else} \end{cases}$$

$$f_2(s_i) = \begin{cases} 0 & \text{if } s_i \text{ is in } \{\text{no value, potential value}\} \\ 1 & \text{else} \end{cases}$$

$$f_3(s_i) = \begin{cases} 1 & \text{if } s_i \text{ has high value} \\ 0 & \text{else} \end{cases}$$

Intuitively, f_1 encodes which sentences have a higher value label than 'no value.' Similarly, f_2 encodes which have a higher value than 'potential value' and f_3 does the same for 'certain value.' Sequentially applying the 3 classifiers (each of them trained on one of the transformations) yields the probability distribution over the 4 classes:

$$\begin{aligned} p(s_i = \text{'no value'}) &= 1 - p(s_i > \text{'no value'}) \\ p(s_i = \text{'potential value'}) &= p(s_i > \text{'no value'}) - p(s_i > \text{'potential value'}) \\ p(s_i = \text{'certain value'}) &= p(s_i > \text{'potential value'}) - p(s_i > \text{'certain value'}) \\ p(s_i = \text{'high value'}) &= p(s_i > \text{'certain value'}) \end{aligned}$$

Since this approach respects the ordinal nature of the labels, we expected an increase in classification performance.

As [11] points out, the first major shortcoming of a point-wise approach to ranking is the model's blindness to the fact that the retrieved documents are grouped by the respective queries. Consequently, as the number of documents associated with different queries varies widely, some queries have much more weight than others. Another problem is that the training optimizes for the regression/classification performance, whereas what matters is the relative ordering of the documents [11]. The pair-wise approach mitigates this problem by focusing on the relative order of two documents. Here, the algorithms are

trained to assess a pair of documents and decide which should be ranked higher [11]. The core idea of the pair-wise approach is the following transformation of the data set:

$$\langle \mathbf{X}, \mathbf{y} \rangle = \left\langle \begin{array}{ccc|c} x_{1,1} & \cdots & x_{1,m} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{n,1} & \cdots & x_{n,m} & y_n \end{array} \right\rangle \rightarrow \left\langle \begin{array}{ccc|c} x_{1,1}-x_{i,1} & \cdots & x_{1,m}-x_{i,m} & y_1 > y_i \\ \vdots & \ddots & \vdots & \vdots \\ x_{n,1}-x_{j,1} & \cdots & x_{n,m}-x_{j,m} & y_n > y_j \end{array} \right\rangle$$

Here, $x_{i,j}$ stands for the j-th feature of the i-th document $\vec{x_i} \in \mathbf{X}$. Notice that the number of data points in the data set increases from n to at most $n(n-1)/2$ and that the original labels (continuous, ordinal, binary) become binary (i.e., 'more valuable,' 'less valuable').

For each sentence pair considered, we obtain the probability that the first sentence is more valuable. The contribution of each sentence pair (s_i, s_j) to the aggregate score is:

$$s_i \leftarrow p(s_i > s_j) - p(s_i < s_j)$$
$$s_j \leftarrow p(s_i < s_j) - p(s_i > s_j)$$

Rather than simply considering the number of "wins," as is usually done, this approach takes into account the confidence of the prediction. Consider the following two examples:

$$p(s_i > s_j) = 1.0$$
$$p(s_i > s_j) = .51$$

In both cases, s_i is deemed more valuable. While in the first case the contribution of this comparison is going to be $s_i \leftarrow 1$ and $s_j \leftarrow -1$, the contribution in the second case is only $s_i \leftarrow .02$ and $s_j \leftarrow -.02$. Using this strategy we minimize the influence of data points (sentence pairs) where the classifier has low confidence in its prediction.

6. Results

Table 1 reports the results of the experiments described in Section 5. The first section shows the three baselines' performance. The two regression (in the second section) and the four classification methods (in the third section) trained on the features described in Section 4 all appear to perform better than the three baselines. For SVM, Random Forest, and MLP-based rankers we conclude that they perform significantly better.

The results of the experiments with ordinal classification models are reported in the fourth section of Table 1. Here, the four methods are compared to the SVM multi-label classification model, the most successful one from the previous batch of experiments. Overall, the performance is not better than the base method. Casting the task as an ordinal classification problem does not appear to improve ranking performance.

The results of the experiments with pair-wise models are reported in the bottom section of Table 1. Here, the four pair-wise classification methods are compared to the RF-ORD (Random Forest with ordinal transform) multi-label classification model, which is the most successful one from the previous batches of experiments. Overall, the performance does not seem to be improved over the base method. We conclude that casting the task as pair-wise classification problem does not appear to improve ranking performance.

Table 1. The table shows the results of the experiments with point-wise LTR methods. The NDCG@100 are shown for the small sparse queries (SmSp), small dense queries (SmDs), large sparse queries (LgSp), large dense queries (LgDs), and all of them together (Overall). The bold font indicates statistical significance. Note that the regression and classification models were compared to the three baselines. The models using ordinal classification were compared to the SVM (the best regression/classification model). The models based on the pair-wise approach were compared to the RF-ORD (the best point-wise model).

	SmSp	SmDs	LgSp	LgDs	Overall
POINT-WISE APPROACH					
Random	.67 ± .15	.76 ± .09	.29 ± .18	.48 ± .09	.63 ± .21
BM25	.74 ± .11	.79 ± .11	.37 ± .22	.56 ± .12	.68 ± .20
BM25-c	.76 ± .09	.80 ± .11	.42 ± .17	.55 ± .13	.70 ± .18
LinReg	.77 ± .12	.82 ± .10	.60 ± .08	.61 ± .08	.75 ± .14
AdaBoost	.79 ± .09	.81 ± .11	.61 ± .08	.64 ± .08	.75 ± .13
LogReg	.78 ± .10	.83 ± .10	.57 ± .10	.65 ± .13	.75 ± .14
SVM	.80 ± .11	.84 ± .11	.63 ± .05	.63 ± .13	**.77 ± .13**
RF	.81 ± .11	.83 ± .11	.55 ± .17	.58 ± .15	**.75 ± .17**
MLP	.78 ± .10	.82 ± .09	.59 ± .10	.63 ± .15	**.75 ± .14**
LogReg-ORD	.77 ± .11	.82 ± .10	.64 ± .04	.64 ± .11	.76 ± .12
SVM-ORD	.80 ± .09	.82 ± .09	.61 ± .04	.64 ± .13	.76 ± .13
RF-ORD	.82 ± .10	.83 ± .10	.67 ± .08	.58 ± .14	.77 ± .14
MLP-ORD	.76 ± .14	.82 ± .10	.59 ± .06	.61 ± .15	.74 ± .15
PAIR-WISE APPROACH					
LR-PWT	.79 ± .13	.82 ± .11	.56 ± .10	.65 ± .08	.75 ± .14
SVM-PWT	.80 ± .09	.82 ± .09	.61 ± .04	.64 ± .13	.76 ± .13
RF-PWT	.81 ± .11	.83 ± .10	.68 ± .08	.64 ± .09	.77 ± .12
MLP-PWT	.76 ± .14	.82 ± .10	.59 ± .06	.61 ± .15	.74 ± .15

7. Discussion and Conclusions

Figure 2 provides a per-query overview of the performance of the best LTR system (Random Forest with pair-wise transform) as compared to the three baselines. The bulk of the improvement subsists in correcting the disastrous performance on the queries at the left tail of the swarm plots. Events on the left side dwarf the improvements happening at the right. Presumably, the BM25 and BM25-c baselines cannot take into account certain aspects of the definition of the sentences' usefulness (e.g., provision of extra information over what is known from the source provision or use of the term of interest in a different sense). For example, the "essential step" query has 2,226 'no value' sentences out of the total number of 2,374, i.e., only about 6.2% of the sentences have a higher value. The query is an extreme example of the need to detect that a term is being used in a different sense. The system also needs to detect if more information is provided than that in the source provision. The RF-PWT system handles these issues quite well; the NDCG@100 for this query is well above 0.8. The system's top results confirm this:

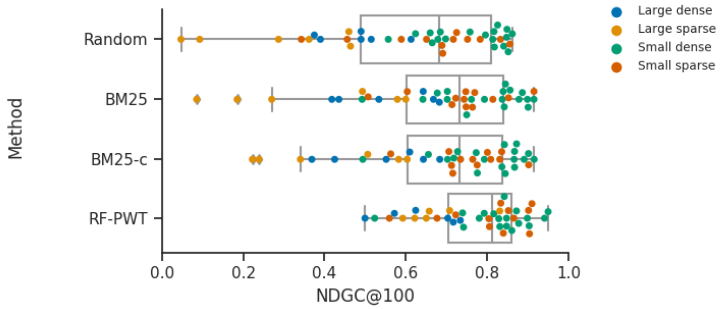

Figure 2. The figure shows scatter plots of the performance on the individual 42 queries measured in terms of NDCG@100. The best performing LTR method (RF-PWT) is compared to the three baselines.

1. That is the import of the phrase "essential step in the utilization of the computer program" that appears in the statute and the phrase "to that extent which will permit its use" that appears in the CONTU Report does not permit a Nibble purchaser to authorize the defendant to put the programs on a disk for him.
2. Even though the copy of Vault's program made by Quaid was not used to prevent the copying of the pro-gram placed on the PROLOK diskette by one of Vault's customers [. . .], and was, indeed, made for the express purpose of devising a means of defeating its protective function, the copy made by Quaid was "created as an essential step in the utilization" of Vault's program.
3. The Sheriffs Department also argued that the copying was an "essential step" under 17 U.S.C. § 117(a)(Z), because the hard drive imaging process was a necessary step of installation.

The first two sentences have 'high value' and the third one has 'certain value.'

Our feature importance study reveals how the LTR systems work. The scores of the two baseline methods (BM25 and BM25-c) are among the most useful features. All the features (except one) encode the relationship between a sentence and either the term of interest or the source provision. The exception is the Sentence/Case Ratio feature; it appears to inform the classifier about an important property of a query. Thus, different queries may require different treatment—an interesting insight for future work.

Despite the improvement of the LTR systems over the baselines, more can be done. Although the most important features show that the LTR systems employ useful strategies to score sentences, the strategies miss some of the information that makes a sentence useful. The methods rely mostly on information about the occurrence counts of the term of interest within a paragraph and the whole decision, on the topical match between the source provision and the decision, and on information about the sentence's novelty with respect to the source provision. From this perspective, pre-trained language models, based on deep neural network architectures, may take additional information into account. We have begun to explore whether such methods can tackle the sentence retrieval problem in a learning-to-rank framework without the need for hand-crafted features.

We have provided evidence that given a statutory provision, a phrase in that provision, and a database of case law, a computer can autonomously rank sentences retrieved from cases by their utility for explaining the meaning of the phrase. Given the set of features indicating sentences' usefulness, retrieving sentences can be tackled as a learning-to-rank problem. We have shown that SVM and Random Forest methods significantly outperform the BM25 and BM25-c baselines. Although casting the task into ordinal or pair-wise relevance classification did not lead to improvements over the multi-class clas-

sification methods, the Random Forest model based on a pair-wise transformation yielded the best performing model overall.

References

[1] Ashley, K. D., and V. R. Walker. "From Information Retrieval (IR) to Argument Retrieval (AR) for Legal Cases: Report on a Baseline Study." *JURIX*. 2013.

[2] Bansal, A., Z. Bu, B. Mishra, S. Wang, K. D. Ashley, and M. Grabmair. "Document Ranking with Citation Information and Oversampling Sentence Classification in the LUIMA Framework." *JURIX*. 2016.

[3] Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[4] Drucker, H. "Improving regressors using boosting techniques." *ICML*. 1997.

[5] Frank, E., and M. Hall. "A simple approach to ordinal classification." *European Conference on Machine Learning*. 2001.

[6] Friedman, M. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance." *Journal of the american statistical association* 32.200 (1937): 675-701.

[7] Grabmair, M., K. D. Ashley, R. Chen, P. Sureshkumar, C. Wang, E. Nyberg, and V. R. Walker. "Introducing LUIMA: an experiment in legal conceptual retrieval of vaccine injury decisions using a UIMA type system and tools." *ICAIL*. 2015.

[8] Holm, S. "A simple sequentially rejective multiple test procedure." *Scand. J. Statistics* (1979): 65-70.

[9] Hyman H., T. Sincich, R. Will, M. Agrawal, B. Padmanabhan, W. Fridy. A process model for information retrieval context learning and knowledge discovery. *Artificial Intelligence and Law* 23(2) (2015):103–132.

[10] Juršic, M., I. Mozetic, T. Erjavec, and N. Lavrac. "Lemmagen: Multilingual lemmatisation with induced ripple-down rules." *Journal of Universal Computer Science* 16.9 (2010): 1190-1214.

[11] Liu, T. *Learning to rank for information retrieval*. Springer Science & Business Media, 2011.

[12] MacCormick, N., and R. Summers. *Interpreting statutes: a comparative study*. Routledge, 2016.

[13] Qin, T., T. Y. Liu, J. Xu, and H. Li. "LETOR: A benchmark collection for research on learning to rank for information retrieval." *Information Retrieval* 13.4 (2010): 346-374.

[14] Saravanan M., B. Ravindran, and S. Raman. Improving legal information retrieval using an ontological framework. *Artificial Intelligence and Law* 17(2) (2011):101–124.

[15] Savelka, J., H. Xu, and K. D. Ashley. "Improving Sentence Retrieval from Case Law for Statutory Interpretation." *ICAIL*. 2019.

[16] Savelka, J., and K. D. Ashley. "Segmenting US Court Decisions into Functional and Issue Specific Parts." *JURIX*. 2018.

[17] Savelka, J., V. R. Walker, M. Grabmair, and K. D. Ashley. Sentence boundary detection in adjudicatory decisions in the united states. *Traitement automatique des langues*, 58, 21. 2017.

[18] Savelka, J., and K. D. Ashley. "Extracting case law sentences for argumentation about the meaning of statutory terms." *ArgMining2016*. 2016.

[19] Savelka, J., and J. Harasta. "Open Texture in Law, Legal Certainty and Logical Analysis of Natural Language." *Logic in the Theory and Practice of Lawmaking*. Springer, Cham, 2015.

[20] Schweighofer, E., and A. Geist. Legal query expansion using ontologies and relevance feedback. *LOAIT* 7 (2007):149—160.

[21] Walker, V. R., P. Bagheri, and A. J. Lauria. "Argumentation Mining from Judicial Decisions: The Attribution Problem and the Need for Legal Discourse Models." *ASAIL Workshop*. 2015.

[22] Walker, V. R., J. H. Han, X. Ni, K. Yoseda. "Semantic types for computational legal reasoning: propositional connectives and sentence roles in the veterans' claims dataset." *ICAIL*. 2017.

[23] Walter S. Definition extraction from court decisions using computational linguistic technology. *Formal Linguistics and Law* 212 (2009):183

[24] The President and Fellows of Harvard University. 2018. *Caselaw access project*. `https://case.law/`. Accessed: 2018-12-21.