Introducing LUIMA: An Experiment in Legal Conceptual Retrieval of Vaccine Injury Decisions using a UIMA Type System and Tools

Matthias Grabmair
Intelligent Systems Program
University of Pittsburgh
mag134@pitt.edu

Kevin D. Ashley
Intelligent Systems Program
School of Law
University of Pittsburgh
ashley@pitt.edu

Ran Chen
Language Technologies Institute
Carnegie Mellon University
ranc@cs.cmu.edu

Preethi Sureshkumar
Language Technologies Institute
Carnegie Mellon University
psureshk@andrew.cmu.edu

Chen Wang
Language Technologies Institute
Carnegie Mellon University
chenwan1@cs.cmu.edu

Eric Nyberg
Language Technologies Institute
Carnegie Mellon University
ehn@cs.cmu.edu

Vern R. Walker
Maurice A. Deane School of Law
Hofstra University
vern.r.walker@hofstra.edu

ABSTRACT

This paper presents first results from a proof of feasibility experiment in conceptual legal document retrieval in a particular domain (involving vaccine injury compensation). The conceptual markup of documents is done automatically using LUIMA, a law-specific semantic extraction toolbox based on the UIMA framework. The system consists of modules for automatic sub-sentence level annotation, machine learning based sentence annotation, basic retrieval using Apache Lucene and a machine learning based reranking of retrieved documents. In a leave-one-out experiment on a limited corpus, the resulting rankings scored higher for most tested queries than baseline rankings created using a commercial full-text legal information system.

Keywords

legal document retrieval, semantic retrieval, natural language processing, argumentation mining

1. INTRODUCTION

In this work, we demonstrate the feasibility of extracting argument-related semantic information from legal texts and using it to improve a full-text legal information system's ranking of retrieved documents. The project was a collaborative effort involving, in no particular order, the Research Laboratory for Law, Logic and Technology (LLT Lab) of the Maurice A. Deane School of Law at Hofstra University, the University of Pittsburgh's Learning Research and Development Center and the Language Technologies Institute at Carnegie Mellon University.

We are working with a set of U.S. Court of Federal Claims cases deciding whether compensation claims comply with a federal statute establishing the National Vaccine Injury Compensation Program. A claimant may obtain compensation if the vaccine caused the injury. To establish causation under the rule of Althen v. Secr. of Health and Human Services, 418 F.3d 1274 (Fed.Cir. 2005), the petitioner must establish by a preponderance of the evidence that: (1) a "medical theory causally connects" the type of vaccine with the type of injury, (2) there was a "logical sequence of cause and effect" between the particular vaccination and the injury, and (3) a "proximate temporal relationship" existed between the vaccination and the injury. The corpus comprises all decisions in a 2-year period applying the *Althen* test of causation-in-fact (35 decision texts, 15-40 pages per decision). In these cases, the Special Masters decide which evidence is relevant to which issues of fact, evaluate the plausibility of evidence, organize evidence and draw reasonable inferences, and make findings of fact. Previously, an underlying argumentation model, the Default Logic Framework [28, 29, 3], has been applied to the cases; it "integrates numerous units of reasoning," each "consisting of one conclusion and one or more immediately supporting reasons" [29].

2. THE LUIMA APPROACH

Our system is based on UIMA, an open-source Apache framework that has been deployed in several large scale government-sponsored and commercial text processing ap-

ICAIL '15, June 08 - 12, 2015, San Diego, CA, USA
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3522-5/15/06...$15.00.
http://dx.doi.org/10.1145/2746090.2746096.

plications, most notably, IBM's Watson question answering system [10]. A UIMA pipeline is an assemblage of integrated text annotators. The annotators are "a scalable set of cooperating software programs, . . . , which assign semantics to some region of text" [13], and "analyze text and produce annotations or assertions about the text" [14, p. 74]. A type system serves as the basis of communication among these annotators; a type system embodies a formalization of the annotators' analysis input and output data [10, p. 3].

Our working hypothesis is that by semantically annotating documents and retrieving them based on the annotations, we can potentially outperform current systems that do not take into account such semantics in their retrieval process. For example, imagine an attorney needs precedent cases concerning whether a tetanus vaccine can cause gastroparesis, so that she may evaluate whether or not to sue on behalf of a concerned client. In her mind, the attorney may ask (Q1) "What is the rule that applies for establishing causation between a vaccine and a subsequent injury?," or (Q2) "Have there in fact been cases where it was held that a tetanus vaccine can cause gastroparesis?" Consider the three following sentences from the *Roper* decision:

S1: *"[T]he petitioner must supply 'proof of a logical sequence of cause and effect showing that the vaccination was the reason for the injury;' the logical sequence must be supported by 'reputable medical or scientific explanation, i.e., by evidence in the form of scientific studies or expert medical testimony.' Althen, 418 F. 3d at 1278."*

S2: *"Dr. Caserta argued that there simply is not enough evidence upon which to reasonably base a conclusion that petitioner's tetanus vaccination caused her chronic gastroparesis."*

S3: *"In this case, I conclude that there exists sufficient evidence in the record to conclude that it is 'more probable than not' that petitioner's chronic gastroparesis was caused by her tetanus vaccination."*

All three sentences are about causation, vaccines and injuries, but with significant differences. S1 states a legal rule and would be the most relevant piece of text if the attorney were to enter her first question Q1 into a legal search engine. To retrieve S1 as most relevant, however, the system would need to recognize the concept of a legal rule, vaccinations, causation and injury beyond the mere textual match. S1 speaks of the vaccination "being the reason for" the injury, which is a an alternative way of referring to, or *mentioning*, the causation concept. Also, it does not mention the term "rule" or "standard" but instead states that the "petitioner must supply proof". Vaccination and injury are mentioned verbatim, but might just as well be phrased as "immunization" and "adverse medical condition," respectively.

Similarly, the attorney would like to see the conclusion in S3 retrieved as the most relevant result when entering Q2 into a search engine. However, if the recapitulation of the expert witness's testimony in S2 were to be retrieved on top, it would be less useful to her inquiry and possibly even misleading because in the same decision the special master disagrees with Dr. Caserta and concludes with S3. These two sentences share the same formulations of "tetanus vaccination", "caused" and "chronic gastroparesis". The negative polarity and the attribution of the statement to Dr. Caserta should flag S2 as not being an evidence-based conclusion by the special master. By the same token, S3 appears as such a conclusion because of the "I conclude" formulation.

In this example, to serve the attorney in a satisfactory way, a legal document retrieval engine needs to be familiar with the possible ways to express the concepts of vaccines, causation and injuries in the English language and be able to identify what type of argumentative purpose the sentence fulfills. Automating this mapping of unstructured terms and formulations as found in legal documents into a type system and using this conceptual understanding to respond to user queries is the core goal of this project.

We have developed an experimental system pipeline comprising a basic type system, automated text annotators and a search as well as reranking component. The pipeline extends the whole way from unstructured text to responding to conceptual queries. Since it centers around UIMA technology, we label it *LUIMA*, for "Legal UIMA". In this paper, we present this system as a proof of feasibility for such an undertaking and for the potential utility of pursuing research in semantic legal information retrieval. Ultimately, we plan to release LUIMA as open source software to leverage future research efforts by the AI&Law community.

2.1 Outline of the Experiment

To evaluate the hypothesis, we conducted an experiment in conceptual legal document retrieval. A schematic outline of the experiment from data to evaluation is given in Fig. 1. We created eleven natural language queries (see Fig. 2) in the vaccine injury domain, focusing on certain legal concepts (vaccines, injuries, causation) occurring in two contexts (statements of rules and statements of evidence based findings). Each query was translated into a *baseline query* and a *LUIMA query* for our experimental system. The baseline queries were entered into WestlawNext^{TM}(WLN) and the top 30 ranked documents for each query were recorded as *baseline ranks* and pooled as the document base. This resulted in a total of 188 documents. The top 30 ranked documents for each posed query were assessed by a legal expert as to their usefulness and reranked to form the *true rank* of the document for a given query.

Of the eleven WLN queries, ten were derived from the facts of particular cases, which the queries were expected to retrieve. These ten *gold standard cases* were manually annotated with the LUIMA sentence types. Three of them were perused to manually craft rule-based annotators to autonomously annotate terms, mentions and formulations (i.e. annotations that span one to several words inside a sentence). Also, the gold standard cases were used to train a classifier which uses the sub-sentence annotations, along with n-gram features, to predict the sentence-level annotations of all non-gold-standard documents in the pool. As the resulting document corpus and query set are very limited, we did not set aside a dedicated test set of queries and documents but rather evaluated our system using leave-one-out cross-validation in both annotation and retrieval tasks, which we deemed appropriate for this feasibility experiment.

The first step in the pipeline is called *LUIMA-Annotate* (see Fig. 1). It uses said rule-based and machine learning annotators to mark up each document. The second step is the experimental search system named *LUIMA-Search*. It enters all documents into a database index using the sentence- and sub-sentence-level annotations. When receiving a query, the system retrieves and ranks its top 30 documents. In a final step, called *LUIMA-ReRank*, we trained a reranking model using the true ranks and features extracted from the

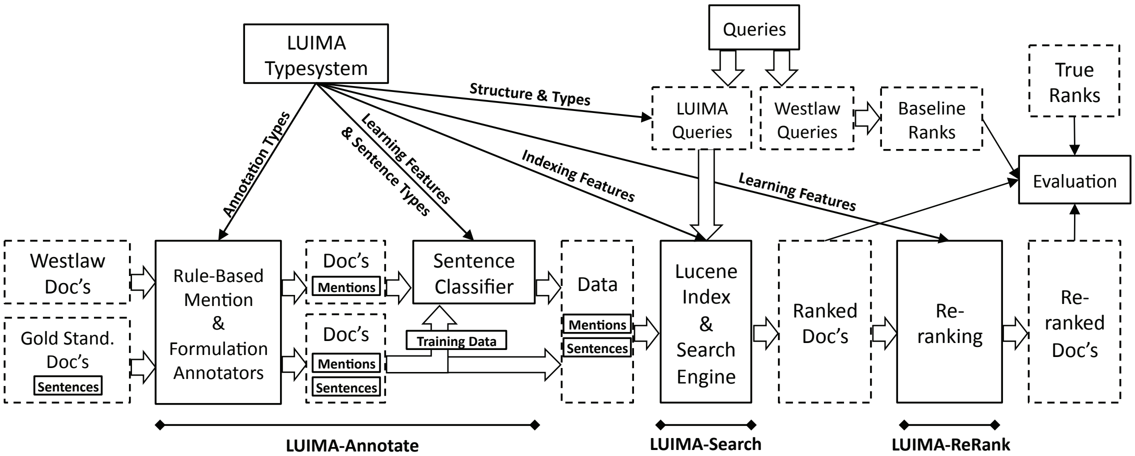

Figure 1: LUIMA Pipeline Architecture. Process modules have bold outlines, data collections dashed.

query and search engine ranking. The documents retrieved by LUIMA-Search are then reranked using this model and the new ranking is evaluated against the search engine's original ranking as well as against the true and baseline ranks.

2.2 The Type System

A UIMA type system is a graph of concepts that relate to each other in various ways as subtypes, supertypes and attribute types. It functions as an ontology for the system by defining the kinds of annotations that can occur in a document, as well as their relationships.

The LLT Lab has developed a sentence-level type system for argumentation in judicial decisions, with which it classifies the principal argumentation roles of sentences or clauses. The nine types identify sentences that primarily state one of the following: citation, legal rule, legal policy, policy-based reasoning, legal ruling or holding of law, evidence-based finding of fact, evidence-based intermediate reasoning, evidence, and case-specific process or procedural facts.

The following two types were used in this experiment: The type *LegalRuleSentence* identifies sentences that primarily state a legal rule in the abstract, without applying it to the particular case. The type *EvidenceBasedFindingSentence* identifies sentences that primarily report a factfinder's finding on whether or not the evidence in a particular case proves that a rule condition has been satisfied.

Sub-sentence annotations are divided into three levels of gradually increasing specificity. The lowest level are Term annotations, which are essentially dictionary annotators and contain types such as *PlaintiffTerm* (which includes all words which can refer to the Plaintiff or Petitioner of the case), *IllnessTerm* or *VaccineTerm*. The second level are Mention types, which are annotated with rules that use Term types and further language clues. For example, a *VaccineMention* or *VaccinationMention* are compositions of a *VaccineAcronym* paired with a *VaccineTerm* ("MMR Vaccine") and *VaccinationTerm* ("MMR vaccination"). The third layer consists of Formulation types, which are small composite structures and flag typical reasoning patterns in legal documents. For example, the phrase "the Plaintiff bears the burden of showing that" will be annotated as a *LegalStandardFormulation*. The annotation rule will detect these and similar sentences by chaining a *PlaintiffMention*, a class of expressions synonymous to "bear the burden" and a class of verbs signaling evidence production (e.g., show, produce, establish, ...). Eventually, these annotations are used in the retrieval process and their presence or absence in a sentence is used as binary features in sentence classification.

3. THE DATA

3.1 Gold Standard Annotated Cases

We constructed the gold standard set of annotated cases from judicial decisions from the LLT Lab's Vaccine/Injury Project (V/IP) Corpus, which consists of decisions on claims for compensation under the National Vaccine Injury Compensation Program (VICP) in the United States [29]. We selected the same ten cases employed in a previous study [2]: *Cusati*, *Casey*, *Werderitsh*, *Stewart*, *Roper*, *Walton*, *Thomas*, *Meyers*, *Sawyer*, and *Wolfe* (case citations on file with authors). The full process for curating a gold standard, annotated decision involves three levels: (1) a single researcher trained both in law and in the sentence-level type system initially marks up the decision; (2) a second trained researcher reviews these annotations, and the first and second reviewers resolve any differences in annotation; and (3) a law professor reviews these annotations and determines the gold standard annotation. For purposes of this experiment, we annotated only two sentence types (LegalRuleSentence, and EvidenceBasedFindingSentence), and we put selected cases through different levels of the full curation process.

3.2 Search Queries

We conducted eleven searches using WLN. One was designed to find legal rules concerning vaccines, injuries, and causation. Each of the remaining ten queries was based on one of the ten vaccine injury cases, five of which were won by the plaintiff and the other five by the defendant (the government). Each of these queries was designed to retrieve cases involving factual findings or conclusions that would be relevant to creating a legal argument about the facts in that case. We assume that most attorneys, in seeking evidence or arguments related to proving or disproving *Althen* Condition 1 (asserting a medical theory connection), would search for cases where the court made findings or conclusions that the same type of vaccine (Covered-vaccine) can or cannot cause the same type of injury (Generic-injury) as in the fact scenario. For each of the ten "source cases", shown in Fig. 2, we instantiated the query by substituting values from that case for Covered-vaccine and Generic-injury into a template:

> Qn. finding or conclusion that ⟨Covered-vaccine⟩ can cause [or causes] ⟨Generic-injury⟩

For instance, Q11 for *Casey* was: "finding or conclusion that Varicella vaccine can cause encephalomyeloneuritis".

WLN's search service is the successor system to Westlaw Classic described in [26]. Its retrieval component also goes beyond the evidence derived from frequency information in the documents' texts and reranks the results using expert-generated annotations (e.g., related to West's Key Number System), citation networks of citing and cited sources, and information about documents' popularity and usage given aggregate information from previous users' queries. The ranking function is optimized using machine learning to determine the weights to ascribe to the different features. "In other words, the reranking portion of the machine learns how to weigh the 'features' representing this evidence in a manner that will produce the best (i.e., highest precision) ranking of the documents retrieved." [19]. In addition, information about legal issues deemed relevant to the query is used to recommend documents on related issues.

WLN generates a "Results List" for a query in order of relevance. For each case in the list, it generates its title, court, jurisdiction, date, and citation as well as a "Case Report". The report has two main parts: a) a two-sentence summary of the claim and the court's decision and b) (if the user selects the "Most Detail" option, as we did) four brief excerpts from the case text with the search terms highlighted.

For each of the eleven queries, for each Results List returned by WLN, the second author who is an attorney read WLN's top thirty Case Reports (not the full decisions) and, based on the Case Report, determined if the case was relevant. Regarding the first query, this meant determining if the Case Report reported a legal rule about vaccines causing injuries. Regarding the remaining ten queries, this meant determining if the Case Reports related a finding or conclusion about the specific vaccine causing a specific type of injury, thus indicating the case's likely utility in making an argument about such a scenario. As an indicator of a query's effectiveness in retrieving relevant cases, the attorney noted whether the query retrieved the source case on which the query was based. Finally, the attorney ranked the Case Reports in terms of how relevant the case as described was.

The process resulted in two rankings of WLN's top 30 Case Reports for each query: WLN's ranking, which becomes the baseline ranks, and the attorney's ranking in terms of suitableness for making a legal argument, which becomes the true ranking. The attorney flagged each case in the true ranking with a binary value as to whether it was relevant or not. An overview of the queries and rankings is given in Fig. 2. Some queries (e.g., Q3, Q4) have just one relevant case with the respective vaccine and injury, whereas others (e.g., Q7, Q9) have more relevant cases.

4. THE EXPERIMENT

4.1 Sentence Splitting

The system needs to correctly identify sentence bounds to later automatically mark up sentence level annotations. Many common sentence splitters perform poorly on legal documents. These typically contain lots of abbreviations and case citations using periods that may easily be confused with sentence ends, negatively impacting subsequent machine learning. Lingpipe[1] performs well on common period phenomena (e.g., *Inc.* for incorporation) and we developed an additional module handling law-specific periods (e.g., *v.* for versus). For lack of resources, we did not evaluate the correctness of the module and leave it for future work.

4.2 Automatic Sub-Sentence Annotation

The rule based annotators in the LUIMA-Annotate component are programmed in UIMA Ruta [17], which is a text-matching-based rule language for rapidly developing UIMA annotators. Our rulebase contains seven dictionary annotators (including vaccine abbreviations[2]) and 49 rules of varying degrees of complexity. Some of these are intuitively appealing context independent rules. Additional rules have been authored by working through parts of three V/IP decisions (*Roper*, *Cusati* and *Thomas*), manually extracting terms, mentions and formulations, and crafting them into Ruta rules. In doing so, the first author expanded the scope of the rules intuitively by anticipating variations in wording and structure. The gold standard sentence-level annotation of the three documents was not consulted in this process. As an example, the following Ruta rule detects an instance of mentioning the plaintiff, an instance of a term expressing an obligation, an optional "also" and up to three alternative verbs and assigns a formulation annotation to the text span.

```
(PlaintiffMention MustRelationTerm “also”?
(“prove” | “show” | “establish”))
-> MARK(LegalStandardFormulation);
```

At the time of the final experiment before submission, the type system consisted of eight term types, 14 mention types and 13 formulation types. As LUIMA is under ongoing development, not all rules and types are directly relevant to the annotation process for this experiment.

4.3 Sentence Level Classification

In LUIMA-Annotate, split sentences are classified using three labels: (1) Instance of LegalRuleSentence, (2) instance of EvidenceBasedFindingSentence, or (3) not being annotated. A sentence which is predicted as not being annotated is still considered "labeled". The complete dataset contained 5909 sentences, including 82 instances of EvidenceBasedFindingSentence and 227 instances of LegalRuleSentence. The feature vector comprised all token n-grams up to n=4. Feature values are numerical and represent the n-gram's TF/IDF value, which is a measure of relative predictiveness of a text feature, commonly used in natural language processing. In a second round of experiments, we enriched this vector by a set of features corresponding to certain sub-sentence LUIMA types. Each of these features is a binary variable representing whether the sentence to be represented subsumed an annotation of the respective type.

As an illustration of the features, consider the following sentence: "Dr. Winston concluded that *petitioner* was suffering from *gastroparesis*, a disorder of delayed stomach

[1] Alias-i. Lingpipe 4.1.0. http://alias-i.com/lingpipe, 2008.

[2] from: http://www.cdc.gov/vaccines/about/terms/vacc-abbrev.htm

	Query	Case Name (date) Winner [Althen 1 issue]	# cases returned by WLN	# cases expert deemed relevant in WLN top 30	source case rank in WLN top 30
Q1	legal rule about vaccines causing injury	NA	157	25/30	NA
Q2	finding or conclusion that MMR vaccine causes intractable seizure disorder	Cusati (9/22/05) Pet. [Pet.]	76	9/30	11th
Q3	finding or conclusion that Tetanus vaccine causes chronic gastroparesis	Roper (12/9/05) Pet. [Pet.]	75	1/30	21st
Q4	finding or conclusion that DTaP vaccine causes diabetes	Meyers (5/22/06) Govt. [Govt.]	75	1/30	9th
Q5	finding or conclusion that Tetanus vaccine causes hand, wrist and arm injuries	Sawyer (6/22/06) Govt. [Govt.]	75	0/30	not in top 30 (37th)
Q6	finding or conclusion that Hepatitis A vaccine can cause cerebellar ataxia	Stewart (3/19/07) Pet. [Pet.]	75	1/30	7th
Q7	finding or conclusion that DPT vaccine can cause acute encephalopathy and death	Thomas (1/23/07) Govt. [Govt.]	78	22/30	not in top 30
Q8	finding or conclusion that MMR vaccine can cause myocarditis	Walton (4/30/07) Govt. [Govt.]	76	2/40	16th
Q9	finding or conclusion that Hepatitis B vaccine can cause multiple sclerosis or MS	Werderitsh (5/26/06) Pet. [Pet.]	77	17/30	1st
Q10	finding or conclusion that Hepatitis B vaccine can cause intractable seizure disorder	Wolfe (11/9/06) Govt. [Govt.]	75	4/30	22d
Q11	finding or conclusion that Varicella vaccine can cause encephalomyeloneuritis	Casey (12/12/05) Pet. [Pet.]	75	1/30	7th

Figure 2: Eleven Queries Submitted to WestlawNext

emptying.” The italicized terms will be annotated as PlaintiffTerm and IllnessTerm, respectively. The feature vector for this sentence will consist of TF/IDF frequency information of all possible one-element (e.g. “Dr.”), two-element (e.g., “Dr. Winston”), three-element (e.g., “Dr. Winston concluded”) and four-element (e.g. “Dr. Winston concluded that”) word sequences contained in the sentence and values of 1 for PlaintiffTerm and IllnessTerm.

The experiment was conducted using leave-one-out cross-validation. This means that ten runs were conducted. In each run, a different one of the ten gold standard annotated documents was used as the test set of sentences and the remaining nine documents were used as training data. Performance metric values were computed for every run and averaged (see Fig. 3). We trained Naive Bayes, logistic regression and decision tree models using the Stanford Parser Tools[3] and Weka package.[4]

4.3.1 Classification Performance Metrics

We use the standard precision, recall, accuracy and F-measures to evaluate the sentence classification component.

Accuracy is defined as:

$$\text{Accuracy} = \frac{|\text{correctly labeled sentences}|}{|\text{all labeled sentences}|}$$

Since only relatively few sentences have been annotated with a LUIMA sentence type, a classifier could easily achieve high accuracy by labeling every sentence as not being annotated. To better evaluate the performance of sentence classification, precision and recall are calculated.

Precision is defined as:

$$\text{Precision} = \frac{|\text{sentences annotated w/ correct LUIMA type}|}{|\text{all sentences annotated w/ a LUIMA type}|}$$

where sentences not labeled as a LUIMA type are excluded.

Recall is defined as:

$$\text{Recall} = \frac{|\text{sentences annotated w/ correct LUIMA type}|}{|\text{sentences annot'd w/ that LUIMA type in gold standard}|}$$

The **F-1 score** is defined as the harmonic mean:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Micro F-1 is based on the labeling statistics across all experiments to calculate the score with the overall precision and recall, while **Macro F-1** is the average of the precision and recall from each individual test in the cross validation.

4.3.2 Classification Results & Discussion

We ran each machine learning algorithm on two feature sets. In the first run, the feature vector of a sentence would only consist of its n-gram representation. In the second run, the binary features corresponding to the sub-sentence level LUIMA annotations were added. As shown in Fig. 3, logistic regression and the decision tree classifier outperform Naive Bayes, which scores best in recall but at the cost of low precision. Logistic regression models produce the best F-score at minimal loss of accuracy when compared to decision trees. We hence used the logistic regression models to automatically annotate sentences in all documents in the pool not belonging to the gold standard.

The addition of the LUIMA type features has very little visible effect on classification. This is probably due to the

[3] Finkel, J., Rafferty, A., Kleeman, A., and Manning, C., Stanford Classifier. http://nlp.stanford.edu/software/classifier.shtml, 2003-2014.

[4] http://www.cs.waikato.ac.nz/ml/weka/

	Precision	Recall	Micro F1	Macro F1	Accuracy
Naive Bayes / NGram	0.15	**0.75**	0.13	0.14	0.88
Naive Bayes / NGram + Type	0.16	**0.75**	0.13	0.15	0.89
Decision Tree / NGram	0.53	0.28	0.18	0.23	**0.97**
Decision Tree / NGram + Type	0.53	0.29	0.19	0.23	**0.97**
Logistic Regression / NGram	**0.66**	0.38	**0.24**	**0.31**	0.96
Logistic Regression / NGram + Type	**0.66**	0.38	**0.24**	**0.31**	0.96

Figure 3: Sentence classification performance measurements. Best values printed in boldface.

fact that the majority of rule annotator patterns is likely to be at most of the same length as a four element n-gram. As this project progresses and more sub-sentence types are added, we intend to periodically re-run this classification experiment to observe any effect. Also, the general performance level is rather low, which is likely due to the relative simplicity of the features and the small dataset, both of which we intend to remedy in future work.

4.4 LUIMA-Search

4.4.1 Search Index Creation

We use Apache Lucene[5] as the main search engine in the LUIMA-Search component. It receives the pool of annotated documents, each of which is segmented into sentences and has sentence level and sub-sentence level LUIMA annotations. All documents are then processed into an XML index file which becomes the central database. However, Lucene is traditionally targeted at delivering whole documents and not portions of documents. Hence, Lucene's index usually contains one document entry per actual text document. As we are interested in retrieving sentences from all documents in the pool, we adapt the representation so that every sentence in our dataset becomes a document element for purposes of creating the index file. Each such element has a content attribute containing the sentence's text and a set of other attributes, among them the LUIMA annotations contained in the sentence. Fig. 4 shows an example entry.

The entry of the *id* field comprises the case document identifier from which the sentence stems and the sentence identifier. The *about* field lists the annotations contained within the sentence. This conception of about-ness is our working model but is intended to be refined in future work.

4.4.2 LUIMA Query Representation

The queries given to the experimental system are manually translated into Lucene format, thereby adding references to LUIMA types to the textual representation. For example, Q3's natural language formulation is: *Retrieve all sentences containing a finding or conclusion that a Tetanus vaccination causes chronic gastroparesis.*

The "conventional" baseline query given to WLN for Q3 is: *finding or conclusion that Tetanus vaccine causes chronic gastroparesis.* While superficially similar, the "finding or conclusion" will be operationalized differently in our experimental system. A purely text-based search engine will look for "finding", "conclusion" and "vaccine" as keywords. By contrast, our system understands them as legal concepts requiring an annotation of the corresponding LUIMA type.

When expanded into a LUIMA query, the query becomes: *type: EvidenceBasedFindingSentence, or about: VaccineMention, or about: VaccinationEventMention, or content: "Tetanus vaccine", or about: CausationMention, or about: IllnessMention, or content:"chronic gastroparesis"*

The query becomes a list of the conditions that need to be satisfied in a sentence for it to be considered a match. The field identifiers correspond to the ones in the example sentence representation in Fig. 4. They comprise the type of sentence searched for, the LUIMA annotation types of interest (VaccineMention, VaccinationEventMention, IllnessMention, CausationMention) and two textual matching tokens for the content field ("Tetanus vaccine" and "chronic gastroparesis"). All these query specifications are connected by logical OR connectors. This does not strictly correspond to the conjunctive query formulation, but allows Lucene to retrieve sentences that are only partial matches and rank them according to the number of satisfied query conditions.

At present, we do not have a LUIMA component for automated query expansion from natural language and leave the development of such a tool and interface to future work.

4.4.3 Document Retrieval & Ranking

Upon receiving the query, the system retrieves all sentences that are responsive to the query ranked by each sentence's Lucene score. The score is a function of the degree of the match relative to the query and provided by Apache Lucene's package (i.e. not a custom function developed by us). The document ranking is determined, in descending order, by counting the number of sentences that have been retrieved from each document. Documents, none of whose sentences have been retrieved, are given a rank of zero and appended to the end of the ranking in no particular order.

4.5 LUIMA-ReRank

This first ranking of documents would likely appear flawed to a legal expert; documents should be ranked not only based on the quantity of matching sentences in them, but in terms of the quality of the matches. Relying on the best qualitative match only, however, may produce a brittle system in case one or more documents were to contain a sentence that is very similar to the query but actually irrelevant. Hence, we employed a technique called *learning to rank* to improve the ranking. It stems from the field of question answering but is increasingly being applied in generic information retrieval [5, 6]. The goal of LUIMA's reranking component is to rearrange the search results obtained from LUIMA-Search to bring the relevant results to the top by using a machine learning approach. It learns the weights of a set of features from the true rankings created by the legal expert.

4.5.1 Reranking Features

The following document features were used to learn how to rerank the documents as retrieved by LUIMA-Search:

- Sentence Count: The number of responsive sentences in a given document, i.e. the same number that LUIMA-

[5] http://lucene.apache.org

```
<doc>
<field name="id"> 21 Cl.Ct. 651:-210032610 </field>
<field name="title"> Carter v. Secretary of Dept. of Health and Human Services </field>
<field name="content"> Therefore, the petitioner was required to prove to the Special Master by a preponderance of the
evidence that the rubella vaccine inoculation was the cause in-fact of her JRA. </field>
<field name="level"> sentence </field>
<field name="type"> LegalRuleSentence </field>
<field name="about"> CausationTerm CausationMention PlaintiffTerm VaccineTerm VaccineMention </field>
</doc>
```

Figure 4: Example entry of a sentence in the Lucene index file.

Search used to compute its initial ranking.

- Lucene Score: The highest Lucene score of all sentences in the given document.
- VSS: The maximum cosine vector space similarity value of the about fields of all sentences in a given document and the query.

4.5.2 *Learning the Reranking Model*

Similar to sentence classification in LUIMA-Annotate, the reranking models were created using leave-one-out cross-validation. The eleven queries (along with each document's reranking features, and their true rankings) were organized into eleven folds, in each of which a different query would become the test set and the remaining ten the training set.

The system examines each document's reranking features and learns a logistic regression formula that assigns weights to each reranking feature and computes a new ranking score for a document. In the learning process, the document's true rank for a query is examined and the weights are set so that a global error function is minimized. The completed formula is then used to predict the test query ranking in each fold; evaluation metric values are calculated and averaged.

The reranking models were trained using the Weka logistic regression algorithms using a pairwise approach.

5. SEARCH & RERANK RESULTS

We evaluated our search system by comparing the ranking of the retrieved cases for each query, along with the baseline ranking, to the true ranking created by the legal expert. We did this for four experimental system configurations:

- *Baseline*: WLN's ranking for the queries in Fig. 2.
- *LUIMA-Search*: Search on full LUIMA query, rank documents by counting their retrieved sentences.
- *LUIMA-Search+ReRank*: Search on full LUIMA query, rank documents by counting their retrieved sentences and rerank them using reranking features.
- *LUIMA-Search, no sentence type*: Search on LUIMA query disregarding the sentence type, rank documents by counting their retrieved sentences.
- *LUIMA-Search+ReRank, no sentence type*: Search on LUIMA query disregarding the sentence type, rank documents by counting their retrieved sentences and rerank them using reranking features.

The intuition underlying disregarding the sentence type is the following: For the documents not belonging to the gold standard, sentence level annotation is done by our classification module which, as seen in section 4.3.2, is suboptimal. In order to assess the sentence type classification's possible negative impact on the search component's performance, we examine how the system performs without it.

5.1 Performance Metrics

The goal of our system is to use conceptual retrieval to improve upon the ranking of the documents. To this end, we choose average precision (AP) and normalized discounted cumulative gain (NDCG) as measurements for each individual search, and mean average precision and average NDCG as the measurement for the system as a whole, all of which are normalized in the [0,1] interval with 1 being the best score. Both metrics are common in evaluating ranking performance, albeit being based on different intuitions.

The true rank was created by reranking the cases retrieved by WLN, so the recall (i.e. coverage of all relevant documents in response to a query) of every baseline result is 1. Also, LUIMA-Search appends all non-responsive documents to the end of retrieved cases. Hence, we do not compare our experimental system to the baseline in terms of recall.

Average Precision (AP) is calculated from the top i ranks (in our case in the $[1, 30]$ interval), calculating ordinary precision (i.e. proportion of relevant cases) and averaging over every such "precision at i":

$$AP = \frac{\sum_{i \in R} P@i}{|R|}$$

where R is {positions of the relevant documents}.

Normalized Discounted Cumulative Gain (NDCG) is defined as follows, where $relevance_i \in \{0, 1\}$:

$$NDCG = \frac{DCG}{IDCG} \qquad DCG_p = \sum_{i=1}^{p} \frac{2^{relevance_i} - 1}{log_2\,(i+1)}$$

where p is the rank position, in our case 30. DCG is calculated using the predicted ranking. The "ideal" DCG (IDCG) is calculated using the true ranking. The underlying intuition is that each relevant document contributes to the overall quality of a ranking depending on where it is ranked relative to the ideal ranking.

Mean Average Precision (MAP) is defined as:

$$MAP(Q) = \frac{\sum_{i}^{|Q|} AP_i}{|Q|}$$

where Q is the set of all queries and AP$_i$ is the average precision for each query.

Average NDCG is defined as:

$$averageNDCG = \frac{\sum_{i}^{|Q|} NDCG_i}{|Q|}$$

where NDCG$_i$ is the NDCG for each query.

5.2 Search Performance

The search engine performance evaluation can be seen in Fig. 5. The LUIMA system outperforms the baseline in

	Measure	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Ave.
Baseline	AP	0.85	**1.00**	0.05	0.11	0.03	0.14	0.82	0.09	0.58	0.13	0.14	0.39
	NDCG	0.96	**1.00**	0.22	0.30	0.19	0.33	0.93	0.29	0.83	0.36	0.25	0.54
LUIMA-Search	AP	0.87	**1.00**	0.08	0.13	0.05	**1.00**	0.76	0.18	0.73	0.14	**0.50**	0.52
	NDCG	0.90	**1.00**	0.26	0.32	0.23	**1.00**	0.86	0.41	0.91	0.38	**0.48**	0.64
LUIMA-Search+ReRank	AP	**0.95**	**1.00**	**0.10**	**1.00**	0.08	**1.00**	0.80	0.21	**0.83**	**0.56**	0.10	**0.60**
	NDCG	**0.99**	**1.00**	**0.29**	**1.00**	0.27	**1.00**	0.93	0.44	**0.95**	**0.72**	0.22	**0.71**
LUIMA-Search no sentence label	AP	0.88	**1.00**	0.07	0.08	0.05	**1.00**	0.77	0.19	0.71	0.14	**0.50**	0.52
	NDCG	0.90	**1.00**	0.26	0.26	0.23	**1.00**	0.86	0.41	0.90	0.38	**0.48**	0.64
LUIMA-Search+ReRank no sentence label	AP	0.92	**1.00**	0.06	0.20	**0.50**	0.17	**0.88**	**0.56**	0.74	0.41	0.33	0.52
	NDCG	0.98	**1.00**	0.24	0.39	**0.63**	0.36	**0.97**	**0.76**	0.91	0.69	0.38	0.66
Best LUIMA vs. Baseline	AP	+.1	+.0	+.05	+.89	+.47	+.86	+.06	+.47	+.25	+.43	+.36	+.21
	NDCG	+.03	+.0	+.07	+.7	+.44	+.67	+.04	+.47	+.12	+.36	+.23	+.17

Figure 5: Performance measurements for search and rerank components. Best values are printed in boldface.

all queries but Q2, for which it scores equally well. The magnitudes of the LUIMA system advantages are shown in the last line of 5 (Best LUIMA vs. Baseline).

Q2 is a tie across all conditions, meaning that all systems retrieved the target case at the top of the list. In six of the remaining ten queries (Q1, Q3, Q4, Q6, Q9 and Q10), our experimental retrieval system using sentence types and reranking performed best (LUIMA-Search+ReRank). This configuration also scored the highest overall average.

In another three queries (Q5, Q7 and Q8), our system performed best if the sentence type was disregarded (Luima-Search+ReRank, no sentence label). In Q11, LUIMA-Search performed best, but the reranking model ranked the target case lower than LUIMA-Search did by itself. Finally, in Q1 (the only query about the LegalRuleSentence type), LUIMA's improvement over the baseline is very little as the baseline performs very well already.

Queries Q2-Q11 focus on instances of EvidenceBasedFindingSentence. LUIMA outperforms the baseline in all of these queries but Q2, in which it tied. In seven instances, reranking improved performance. The intuition of relaxing the query conditions by disregarding the sentence type because of the suboptimal performance of the sentence classifier proved useful as it improved performance in three queries.

5.3 Feature Analysis

Different subsets of the reranking features were examined to assess which features were most predictive. Results can be seen in Fig. 6, which summarizes the MAP and meanNDCG gain of different feature subsets (lines 3-7) compared to LUIMA-Search (line 2) and the baseline (line 1).

Almost all subsets perform equally well and better than both the baseline and plain LUIMA-Search. The exception is using only sentence count and vector space similarity (line 4). The critical component seems to be the Lucene score feature. Line 6 shows that using only this feature leads to very good performance. However, adding vector space similarity to it (line 5) seems to marginally improve meanNDCG. When using all three features (line 7), this improvement disappears. Hence, LUIMA-ReRank only used VSS and the maximum Lucene score as rerank learning features for the final experiment runs reported in Fig. 5.

6. DISCUSSION

The outcomes of this early feasibility experiment suggest that the automated annotation of instances of sub-sentence LUIMA types (Vaccines, Injuries, Causation and Sentence Types) in the documents and their use in indexing / querying can significantly improve the search and reranking components. This seems to be the case even though the annotations do not improve the performance of the machine learning based automatic sentence level annotator. Performance increases when disregarding the sentence type indicate that the sentence-level annotator's mediocre performance may block the benefit of identifying legal concepts where the argumentative sentence context is known.

Obvious weaknesses of this experiment are the small number of queries and gold standard annotated documents, the small document pool and the lack of evaluation on a dedicated test set of queries untouched during development. The fact that we used three of the gold standard cases to manually craft the sub-sentence annotation rules is another possible confound. On the other hand, semantic retrieval requires corresponding annotation data to be injected via some automated domain expert annotation, thereby inevitably leading to some form of manual knowledge engineering. We wish to explore the question whether this investment will eventually lead to better retrieval systems. Given limited resources for corpus creation, we could not set aside a dedicated test set of queries and hence chose a leave-one-out evaluation to mitigate some of these potential biases.

The resulting performance improvement over the baseline may arguably be attributed to the specificity of the dataset, the hand-crafted annotator rules as well as the fact that WLN was used as-is without any domain-specific qualification beyond WLN's standard features. Still, we wish to present our LUIMA system and its first results as promising evidence of the feasibility of future research on automated semantic analysis of legal documents and its potential utility for enabling legal information retrieval systems to take advantage of argument-related information.

7. FUTURE WORK

The immediate question is whether the semantic annotation and retrieval approach will scale up to settings featuring queries unknown at development time and, eventually, whether and how well the benefits of the annotation transfers across domains. We intend to expand the type system, annotators and query types, first tackling domains that have similar document structure and common types, and gradually to incorporate more diverse material. We plan to create

	Method	MAP	meanNDCG
1	Baseline	0.38	0.54
2	LUIMA-Search with sentence type	0.52	0.64
3	ReRank - Lucene Score & Sentence Count	**0.60**	0.70
4	ReRank - VSS & Sentence Count	0.47	0.60
5	ReRank - Lucene Score & VSS	**0.60**	**0.71**
6	ReRank - Lucene Score only	**0.60**	0.70
7	ReRank - All features	0.59	0.70

Figure 6: Feature analysis for reranking

a larger corpus of queries and baseline data as well as take account of human annotation reliability concerns, which we could not do in this project due to resource limitations. Two tool components yet to be developed are (1) a LUIMA query expander which can work from (possibly semi-formal) natural language to create LUIMA queries, and (2) a practical user interface to view the search results. The latter is of particular interest as we did not evaluate whether the sentences retrieved by LUIMA-Search are in fact informative for the query at hand beyond contributing to the ranking of the document from which they stem. Also, we intend to extend and refine the reranking features.

Maintaining a UIMA type system and corresponding annotators requires considerable ability to analyze legal language as well as integrate new types and rules with existing ones in order to use the semantic scaffolding effectively. For this project, the type system is of manageable size and only three case texts were consulted in the rule drafting process. We anticipate work on the sub-sentence type system and annotators to become more complex as the system grows. Given enough annotated data, the use of machine learning techniques may become feasible. UIMA's architecture as a service-based umbrella framework for different kinds of annotators provides the needed flexibility.

Our long term plan is to develop the LUIMA typesystem, annotators and associated tools into a package providing basic functionality to enhance or jumpstart development of systems in legal information retrieval, text classification, semantic extraction or similar applications. Once sufficiently mature, LUIMA is planned to be released as open source software on Github under an Apache 2 license.

8. RELATED WORK

A milestone in extracting argument-related information from case decisions is [22]. The authors defined an argument as "a set of propositions, all of which are premises except, at most, one, which is a conclusion. Any argument follows an argumentation scheme. . . ." Using machine learning based on manually classified sentences from the Araucaria corpus [23], including court reports, and the ECHR corpus, a document set of legal texts [9], they achieved accuracies of 73% and 80%, respectively, in classifying sentences as propositions in arguments or not based on domain-general features. They also classified argumentative propositions as premises or conclusions, and, for a subset of documents, their manually-constructed rule-based argument grammar generated argument tree structures [22].

Factors, stereotypical fact patterns which strengthen or weaken a side's argument in a legal claim, have been automatically identified in text in [7]. The SMILE+IBP program learned to classify brief case summaries in terms of applicable trade secret law factors [1], analyzed classified new problems input as texts, predicted outcomes, and explained its predictions. [30] presented a scheme for annotating finer grained factor components (i.e., factoroids) with GATE.

Using an argument scheme to assist in representing cases for conceptual legal information retrieval was first explored in [8]. More recently, a variety of approaches for automatic semantic processing of case decision texts for legal IR has been tried, including automatically extracting: case treatment history (e.g., "affirmed", "reversed in part") [11], offenses raised and legal principles applied from criminal cases to generate summaries [27], case holdings [20], and argument schemes from the Araucaria corpus such as argument from example and argument from cause to effect [12].

In [31], a network of "legal issues" is mined from a case law database, each comprising a "statement of belief, opinion, a principle, etc.," containing legal concepts, that has a legal implication (e.g., "Thirteen-year-olds should not own a vehicle") and whose function is as the proposition for which a case is cited. The system in [21] retrieved documents based on queries containing semantic descriptors and indicators of cross-referential relations between documents (e.g., "Which orders talk about abnormally annoying noise and make reference to decrees talking about soundproofing ?").

Other programs have assigned rhetorical roles to case sentences based on manually annotated decisions [24], categorized legal cases by abstract WLN categories (e.g., finance and banking, bankruptcy) [25] or general topics (e.g., exceptional services pension, retirement) [15], and determined the role of a sentence in the case (e.g., as describing the applicable law or the facts) [16].

In an as yet non-legal context, the IBM Debater system employs domain independent techniques to "detect relevant claims" and return its "top predictions for pro claims and con claims" on a topic [18].[6] For instance, upon inputting the topic, "The sale of violent videogames to minors should be banned", Debater scanned millions of Wikipedia articles, extracted and assessed candidate claims, delivered the pro and con claims as an argument about the topic. Debater may be applied to legal domains (*see, e.g.,*[4]).

9. CONCLUSIONS

Our experiment's positive results demonstrate the feasibility of implementing a conceptual legal document retrieval system going from natural language legal documents to retrieval results for a restricted set of documents in the domain of vaccine injury claims. The LUIMA type system pro-

[6] see, e.g., http://finance.yahoo.com/blogs/the-exchange/ibm-unveils-a-computer-than-can-argue-181228620.html. A demo appears at the 45 minute mark: http://io9.com/ibms-watson-can-now-debate-its-opponents-1571837847

vides general and domain focused annotation types specific to legal textual information. Sub-sentence annotations are identified using manually crafted rules while sentence level annotations are added using a machine learning sentence classifier trained on a small set of gold standard annotated documents. Automatic sentence annotation performance is mediocre. Still, by taking these annotations into account in the retrieval and reranking process, our system almost always scores higher than a set of baseline retrieval results obtained using a commercial full-text legal retrieval system on the task of ranking retrieved documents in a leave-one-out experiment. Naturally, this needs to be confirmed on a larger, more diverse corpus and dedicated test set. Despite the limitations, we consider our results encouraging for future work on semantic information retrieval in AI&Law.

10. ACKNOWLEDGMENTS

The authors would like to thank the student annotators at the LLT Lab: Ben Gerson, Andrew Lauria, Cass Sanford and Newtan Thevarajah. We further thank the University of Pittsburgh's University Research Council Multidisciplinary Small Grant Program for its financial support in developing tools for automated analysis of legal documents.

11. REFERENCES

[1] Ashley, K. & Brüninghaus, S., Automatically classifying case texts and predicting outcomes. *Artificial Intelligence and Law*, 2009, 125-165.

[2] Ashley, K., & Walker, V., From Information Retrieval (IR) to Argument Retrieval (AR) for Legal Cases: Report on a Baseline Study. *Proc. Jurix 2013*, 29-38.

[3] Ashley, K. & Walker, V., Toward Constructing Evidence-based Legal Arguments Using Legal Decision Documents and Machine Learning, *ICAIL 2013 Proc.*, ACM, 2013, 176-180.

[4] S. Beck, Emerging Technology Shapes Future of Law. http://www.americanlawyer.com/id=1202664266769/Emerging-Technology-Shapes-Future-of-Law. Accessed: 2014-09-20, 2014.

[5] Bilotti, M. W., Elsas, J., Carbonell, J. & Nyberg, E., Rank learning for factoid question answering with linguistic and semantic constraints, *Proc. of the 19th ACM Int'l Conf. on Information and Knowledge Management*, CIKM '10, ACM, 2010, 459-468.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai & Hang Li, Learning to rank: From pairwise approach to listwise approach, *Proc. ICML 2007*, ACM, 129-136.

[7] Daniels, J. & Rissland, E., Finding Legally Relevant Passages in Case Opinions, *Proc. ICAIL 1997*, 39-46.

[8] Dick, J. & Hirst, G., A Case-Based Representation of Legal Text for Conceptual Retrieval, *Proc. Workshop on Language and Information Processing*, 1991, 93-102.

[9] European Court of Human Rights Reports of Judgments and Decisions, 2014.

[10] Epstein, E.; Schor, M.; Iyer, B.; Lally, A.; Brown, E. & Cwiklik, J., Making Watson Fast, *IBM J. Res. and Dev.* 56.3.4, 2012, 15-1.

[11] Jackson, P.; Al-Kofahi, K.; Tyrrell, A., & Vachher, A., Information Extraction from Case Law and Retrieval of Prior Cases, *Artificial Intelligence*, 150, 2003, 239-290.

[12] Feng, V. W., and Hirst, G., Classifying arguments by scheme, *Proc. NAACL HLT 2011*, Vol. 1, 987-996, ACL.

[13] Ferrucci, D., Introduction to "This is Watson", *IBM J. Res. and Dev.*, IBM, 56, 2012, 1-1.

[14] Ferrucci, D.; Brown, E.; Chu-Carroll, J.; Fan, J.; Gondek, D.; Kalyanpur, A.; Lally, A.; Murdock, J. W.; Nyberg, E.; Prager, J.; Schlaefer, N. & Welty, C. Building Watson: An Overview of the DeepQA Project, *AI Magazine*, 31, 59-79, 2010.

[15] Gonçalves, T., & Quaresma, P. Is Linguistic Information Relevant for the Classification of Legal Texts?, *Proc. ICAIL 2005*, ACM, 2005, 168-176.

[16] Hachey, B. & Grover, C. Extractive Summarisation of Legal Texts, *Art. Int. and Law*, 2006, 14, 305-345.

[17] Kluegl, P., Toepfer, M., Beck, P. D., Fette, G., & Puppe, F., UIMA Ruta: Rapid development of rule-based information extraction applications, *Natural Language Engineering*, 2014, 1-40.

[18] Levy, R.; Bilu, Y.; Hershcovich, D.; Aharoni, E. & Slonim, N., Context Dependent Claim Detection, *Proceedings of COLING 2014*, 2014, 1489-1500.

[19] Lu, Q. & Conrad, J., Next Generation Legal Search - It's Already Here, https://blog.law.cornell.edu/voxpop/2013/03/28/next-generation-legal-search-its-already-here, Accessed May 22, 2015

[20] McCarty, L. T., Deep semantic interpretations of legal texts, *Proc. ICAIL 2007*, 217-224, ACM.

[21] Mimouni, N.; Fernandez, M.; Nazarenko, A.; Bourcier, D. & Salotti, S., A Relational Approach for Information Retrieval on XML Legal Sources, *Network Analysis in Law*, 2014, 169-192.

[22] Mochales, R., and Moens, M.-F., Argumentation mining, *Art. Int. and Law*, Vol. 19, no. 1 (2011): 1-22.

[23] Reed, C. & Rowe, G. Araucaria: Software for Argument Analysis, Diagramming and Representation, *Int'l J. on Artificial Intelligence Tools*, 2004, 13, 983.

[24] Saravanan, M. & Ravindran, B., Identification of Rhetorical Roles for Segmentation and Summarization of a Legal Judgment, *Art. Int. & Law*, Vol. 18, 45-76, Springer, 2010.

[25] Thompson, P., Automatic Categorization of Case Law, *Proc. ICAIL 2001*, ACM, 2001, 70-77.

[26] Turtle, H., Text Retrieval in the Legal World, *Artificial Intelligence and Law*, Kluwer, 1995, 3, 5-54.

[27] Uyttendaele, C., Moens, M.-F. & Dumortier, J., Salomon: Automatic Abstracting of Legal Cases for Effective Access to Court Decisions, *Artificial Intelligence and Law*, Kluwer, 1998, 6, 59-79.

[28] Walker, V.; Carie, N.; DeWitt, C. & Lesh, E., A Framework for the Extraction and Modeling of Fact-Finding Reasoning from Legal Decisions: Lessons from the Vaccine/Injury Project Corpus, *Artificial Intelligence and Law*, 2011, 291-331.

[29] Walker, V.; Vazirova, K. & Sanford, C., Annotating Patterns of Reasoning about Medical Theories of Causation in Vaccine Cases: Toward a Type System for Arguments, *Proc. Workshop Arg. Mining*, ACL, 2014.

[30] Wyner, A. & Peters, W., Lexical Semantics and Expert Legal Knowledge Towards the Identification of Legal Case Factors, *Proc. JURIX 2010*, 127-136.

[31] Zhang, P.; Silver, H.; Wasson, M.; Steiner, D. & Sharma, S., Knowledge Network Based on Legal Issues, *Network Analysis in Law*, 2014, 21-50.