Midterm Report

Chun-fan Liao
University of California, Berkeley
School of Law
ciliao@berkeley.edu

Kaifei Peng
Wuhan University
Hongyi Honor College
pkf123@berkeley.edu

1 Additional Literatures

In (Justeson and Katz, 1995), a grammatical structure is introduced to capture more frequent multiword phrases within a document, which can be implemented using regular expression.

In (Bommarito et al., 2018), *LexPredict* introduces *LexNLP*, an open source package on NLP for legal texts. It claims to be capable of not only managing NLP routines, such as document segmentation or IE, but also building models for word embedding or tagging. Some of our basic NER tasks may be completed with LexNLP, but we found it cannot correctly segment some types of legal terms, such as patent claim citations or case citations, which we need to do on our own using regular expressions.

(Dozier and Haschart, 2000) illustrates the structure of documents in the dataset and implements a person name extraction system, which dataset highly resembles ours (WestLaw). Based on this structure, the system can segment the synopsis of the document into paragraphs with different functions and thus locate which paragraphs the names to be extracted reside. After that, it parses each paragraph and apply rules to those paragraphs to extract names.

2 Data: Federal Circuit Court Cases

We have so far collected 321 patent-related cases of US Court of Appeals for the Federal Circuit (Federal Circuit Court) from the WestLaw database that are particularly related to *prior art*, the previously publicized invention or knowledge. The number of cases in the training set is 257, the number of cases in the development set is 32, and the number of cases in the test set is also 32. We have selected cases which verdicts have a clear inclination toward the plaintiff(s) or the defendant(s), so that the case outcome can be more easily classified.

3 Scopes: Information Extraction Template for Patent Cases

We introduce an IE template for each patent case of Federal Circuit Court that we collected. The objective of our work is to fill out this template to list useful information for attorneys, which may later be integrated to other legal case retrieval systems.

Take *Amazon.com, Inc. v. Barnesandnoble.com, Inc., 239 F.3d 1343 (Fed. Cir. 2001)* as an example, the filled template should be as follows:

1. **District Court Ruling**: In favor of plaintiff
2. **Circuit Court Ruling**: In favor of appellant: Vacated and remanded
3. **Key Phrases**: preliminary injunction, prior art, single action, shopping cart, server system, client system, purchase order, web site
4. **Patent Terms**: 1–Click, single action ordering, client/server environment
5. **Patent Summary**: “Amazon’s patent is directed to a method and system for “single action” ordering of items in a client/server environment such as the Internet.”
6. **District Court**: United States District Court for the Western District of Washington
7. **Plaintiff**: Amazon.com, Inc.
8. **Defendant**: barnesandnoble.com, inc., and barnesandnoble.com llc
9. **Appellant**: barnesandnoble.com, inc., and barnesandnoble.com llc
10. **Appellee**: Amazon.com, Inc.

Those items are self-explanatory. What worth mentioning is the *Key Phrases*, *Patent Terms*, and *Patent Summary*. *Key Phrases* generally list critical legal issues or other essential technical terms involved for the case. *Patent Terms* are the specific terms used to denote the patent technology. *Patent Summary* is the most important sentence(s) in the judge opinion texts to describe the functionality of the patent at issue.

4 Methods and Discussion

4.1 Rule-based Methods

Terms 1-2, 6-10 in our template mentioned in Section 3 are easy to extract with rule-based methods. These terms tend to appear in the *synopsis* of a document, which is relatively structural piece of text. The structure of synopsis of West-Law document is discussed roughly in (Dozier and Haschart, 2000). All *Terms 1-2, 6-10* occur in a specific order. Using paragraph/sentence segmentation, we can segment the text into pieces, such as *Both Parties Paragraph*, *Court Paragraph*, and *Verdict Paragraph*. After locating the terms in these pieces, we apply rules such as regular expressions to these pieces and extract those terms. For instance, *Both Parties* paragraph generally reads *Party I v. Party II* and regular expressions can perfectly detect *Party I* and *Party II*.

We have tried the rule-based methods on our texts and found the *District Court Ruling (term 1)* and *Circuit Court Ruling (term 2)* can both be extracted, which was then used together with roles of plaintiff(s) and defendants(s) in the Federal Circuit Court to predict the verdict of the case using decision tree classification with 100% of accuracy. We are currently working on extracting other terms (*terms 6-10*) with the certainty that those terms can also be extracted.

4.2 Supervised Method

Rule-based methods will not be effective when it comes to extracting *term 3-5* since they do not appear in structural contexts and are usually hidden in tedious explanatory sentences. We believe that incorporating syntactic and semantic analysis to our texts is necessary for detecting these theme-related terms.

We plan to construct a *classifier/regressor* which will be applied to every candidate word. For each word, it classifies/evaluates the confidence whether the word is the *Patent Term (term 4* we are looking for.

Useful features for classifying whether a word w_i is the *Patent Term* might include:

- The word embeddings of itself and its neighboring words, e.g., w_{i-2}, w_{i-1}, w_{i+1}, and w_{i+2}
- The POS/NER tags of itself and its neighboring words
- The word shape of the word (whether it contains the patent indicator "(r)", etc.)
- The TF-IDF value of the word

To do above work, we are planning to use currently existing tools like spacy / gensim / scikit-learn for aiding. The current to-do list is as the following:

- Manually annotate the patent terms and explanatory sentences in our dataset
- Tokenize documents in a such a way that important word shapes (e.g. "(r)" and case numbers like "239 F.3d 1234") are not deconstructed by segmentation
- Use SpaCy for POS/NER tagging
- Use gensim to train word embeddings on our datasets
- Use scikit-learn to train the classifier/regressor

We are now still improving the tokenization method and annotating our datasets. As for tagging step, since SpaCy is not domain-specific and errors do exist, we are searching a way to do additional training so that the tagger can be more eligible for the legal domain.

After that, we can calculate the confidence score for each word and see whether it is the patent term or key phrase that we are searching. Then, we can compute the confidence score for each sentence by averaging the aforementioned confidence scores for every words it contains, instead of using the mean TF-IDF score of the sentence, which we found not promising. Finally, we can evaluate our performance based on the comparison between the estimation and the answer manually annotated by us. The performance of rule-based estimation and supervised-learning estimation can be observed separately.

References

Michael J. Bommarito, Daniel M. Katz, and Eric M. Detterman. 2018. Lexnlp: Natural language processing and information extraction for legal and regulatory texts. *arXiv preprint arXiv:1806.03688*.

Christopher Dozier and Robert Haschart. 2000. Automatic extraction and linking of person names in legal text. In *RIAO*, pages 1305–1321.

John S. Justeson and Slava M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identication in text. *Natural Language Engineering*, 1(1):9–27.