Hidden in Plain Sight
Using the law to summarize the law

Richard Downe
richard@casetext.com

May 18, 2016
Outline

1. Who is Casetext?
2. Introduction to Legal Research
3. Data Science at Casetext
4. Legal Summarization
5. Conclusion
Background

- Y-combinator class of 2013
- Founded by ex-litigators with tech chops
- Built with goal of improving the context and summarization attached to judicial opinions
- Crowdsourcing of secondary content has been a major focus from the beginning
Our Mission

- $10B/year market
- Dominated by duopoly of Westlaw (Thompson Reuters) and Lexis Nexis (Reed Elsevier) and Bloomberg
- High subscription fees have historically limited access to quality research
- Many solos cannot afford
- Casetext is part of recent movement to disrupt
Our Mission

- American judicial system rooted in English common law
- Precedent is principal currency
- Effectively arguing a case depends upon retrieval of relevant precedent, and persuasive presentation of said precedent
- The volume of prior case-law grows with time
Our Mission

- Legal profession depends upon technology to navigate the growing body of prior opinions
- Summarization and context necessary to decide which opinions to read in full
- Judges routinely summarize the opinions they discuss
- How to leverage this trove of data in a way that helps practitioners of the bar?
Our stack has evolved through the years:

- Began life as a Django-based site on heroku
- Site is now a single-page AngularJS app, backed by Firebase
- First attempt at batch processing used **HTTP-forwarding**
  - Created at google
  - Transform/forwarding instructions contained in HTTP headers
  - It never caught on...
- Hadoop is industry standard, but...
  - High learning curve
  - High administration overhead
- We now rely on Spark, running on AWS EMR
Architecture
2 main types of flows

XML Transformation:
- Marshalling various artifacts into a common XML format
- Akoma Ntoso used internally
- Primarily XSLT-based, with python-extension functions(lxml)
NLP Analysis

- Built around Apache UIMA
- JVM Centric (We prefer Scala, but there’s Java and Jython in there, too...)
- We’re language agnostic, but tools seldom are
- Uses combination of off the shelf NLP tools and formal grammars
- Entities are identified, indexed, and analyzed together
Architecture

UIMA

- Originally developed by IBM
- Engine inside the IBM Watson project
- XML-overload addressed by companion project UIMA-Fit
- Core concepts: an analysis engine is composed of one or more annotators
- Annotations stored in efficient in-memory index for rapid type-based traversal
YO DAWG I HEARD YOU LIKE PIPELINES
Architecture

In Pictures
Legal writing is highly formulaic

Deterministic techniques that don't work on general natural language text are effective

Context-free grammars can be used for highly accurate Named Entity Recognition (NER) for certain entities

Accurate deterministic NER can inform downstream analysis

Citations widely recognized example of this ( “372 U.S. 335” – Gideon v Wainwright)
Explanatory Parentheticals

- Recommended by *The Bluebook* (The “Bluebook” is the style manual that dictates much of the structure in legal writing.)
- Typically authored by the deciding judge – so less is lost by third party interpretation
- Concise, and often summarize the holding in a useful way
- Leading present-participle is reliable guide to a summarizing parenthetical’s location
Explanatory Parentheticals

How to harvest?

- Generally follows a citation, which provides a useful anchor
  - e.g. Lewis v. Casey, 518 U.S. 343, 116 S. Ct. 2174, 135 L.Ed.2d 606 (1996) (holding that inmates claiming denial of access to courts failed to show actual injuries stemming from inadequate library facilities).\(^1\)

- Arredondo\(^2\) demonstrated the utility of this technique using regular expressions in *Legal Information Review*

---


\(^2\) Legal Information Review Vol. 1, p.31-49 (2015-16)
I have a hammer...

v\.[^\)]{10,150}[0-9]{4}\]^(?!quoting)(?!citing)(?!concurring)(?!dissenting)(?!emphasis)(?!citation)(?!in\_banc)(?!en\_banc)(?!per\_curiam)(?!per\_cuirum)(?!footnote)(?!\")[a-z][^\)]{2,9}ing[^\%]*?(?<!\^[a-z])(?<!\^[A-Z])(?<!\^[0-9])(?<![0-9]{4})\])\))
A more refined approach

1. Identify all case citations in the text
2. Identify all matching pairs of parens
3. Filter out parentheticals containing disqualifying phrases ("rehearing granted")
4. Attempt to associate remaining parentheticals with a citation
5. Keep only those where association with citation is successful
6. Contents of the parenthetical analyzed for suitability: leading present-participle contains a lot of information.
   - holding
   - finding
   - concluding
   - etc...
Key Passages

- The most heavily quoted portions of a text are likely to be significant.
- They may not summarize, but they likely encapsulate key points of interest.
- Long quotations may be easy to pin down, but how to correctly place shorter quotes?
Key Passages
First Pass

Similar approach!

1. Identify all matching pairs of quotations in text
   - This is much harder than it ought to be
   - There are many typos in the case law
   - Bounding the start and end solves many of the issues

2. Attempt to match quote with a proximal reporter citation

3. If cannot, search invert index for unique match
Key Passages
Second Pass

1. Retrieve all quotations inbound to a case
2. Find first instance of each quotation in text
3. Store contiguous blocks of quoted text as key passages
By the Numbers

- 2479077 unique summaries found in 8 million cases
- 677463 cases summarized with a parenthetical at least once
- 4759799 unique key passages identified
- 1630991 cases summarized with a key passage
The citation graph within the law provides a rich source of summarization.

Questions?

We’re hiring!