Legal Linking: Citation Resolution and Suggestion in Constitutional Law

Robert Shaffer[*]
Perry World House
University of Pennsylvania
shafferr@upenn.edu

Stephen Mayhew[*]
Computer and Information Science
University of Pennsylvania
mayhew@seas.upenn.edu

Abstract

This paper describes a dataset and baseline systems for linking paragraphs from court cases to clauses or amendments in the US Constitution. We implement a rule-based system, a linear model, and a neural architecture for matching pairs of paragraphs, taking training data from online databases in a distantly-supervised fashion. In experiments on a manually-annotated evaluation set, we find that our proposed neural system outperforms a rules-driven baseline. Qualitatively, this performance gap seems largest for abstract or indirect links between documents, which suggests that our system might be useful for answering political science and legal research questions or discovering novel links. We release the dataset along with the manually-annotated evaluation set to foster future work.

1 Introduction

Authors of legal texts are frequently interested in understanding how their document relates to a knowledge base or to some reference text or corpus. Because legal reasoning relies on references to preexisting precedent, identifying the documents or document sections (e.g. court cases; constitutional provisions) that relate to the author's current argument or topic of interest is an important task. However, constructing these reference links is labor-intensive, particularly if the set of reference texts is large or the link is ambiguous. Automating this linkage task therefore offers useful assistance for authors of legal texts.

Linking systems of this kind are also useful for answering important political science and legal research questions. For example, in US Constitutional law, the Supreme Court has anecdotally appeared more receptive to arguments that combine multiple Constitutional rights.[1] However, without an automated linking system, identifying instances of rhetorical "commingling" of rights is labor-intensive. Outside the American context, even simple data on the frequency with which judges invoke particular constitutional rights are difficult to gather. A generic, automated system capable of inferring links between sections of judicial opinions and related legal texts[2] would therefore be valuable for legal and political science researchers.

With this motivation, we present a method for linking pairs of documents – here, Supreme Court case paragraphs and Constitution sections – based on distantly-annotated training data. Our model operates on the level of short pieces of text, such as paragraphs, and gives a binary decision between pairs of texts, marking a presence or absence of a link. As we describe, a key challenge we face is that our training data are generated using rules-based heuristics, and are thus highly incomplete. As a result, one of our main contributions is a data preprocessing step that "strips" rules-based language from the training data. In our experiments, we find that this step combined with a modern neural network model allows our system to substantially outperform both rules-driven and non-neural baselines on a manually-tagged evaluation set. Qualitatively, this performance gap appears largest for paragraphs that contain abstract or indirect references to Constitutional provisions, which suggests that the system we propose might also be useful for discovering new links not identified by existing techniques.

[*]Authors contributed equally. Order chosen by coin flip.

[1]E.g., *Lamb's Chapel v. Center Moriches Union Free School District* (508 U.S. 384) and related religious speech cases, which successfully combine free speech and free exercise of religion arguments. See McCloskey and Levinson (2016, 162-163) for further discussion.

[2]E.g. other national constitutions, as in Elkins et al. (2014).

2 Related Work

In the legal domain, initiatives including the Cornell Legal Information Institute have constructed standardized citation templates to assist users interested in linking citations of various formats (see Casanovas et al. (2016) for an overview). However, these systems are not designed to infer citations based on plain-text excerpts, which is our problem of interest. Schwartz et al. (2015) propose a topic model-based approach that suggests citations to relevant US Supreme Court case law based on user-inputted free text. Because this system only draws links between excerpts and full Supreme Court cases, it is coarser than ours, but provides perhaps the closest point of comparison in the legal domain (Branting, 2017). A closer comparison point is Nomoto (2018), who propose an approach that infers paragraph-level citation links between published scientific papers.

The methods that we adopt to solve this problem draw inspiration from several fields in natural language processing (NLP) and machine learning, including multilabel classification (Boutell et al., 2004; Nam et al., 2014), dataless classification (Chang et al., 2008), citation resolution (Duma and Klein, 2014), and entity linking (Ratinov et al., 2011; Shen et al., 2015). Our method of collecting training data is reminiscent of distant supervision techniques (Mintz et al., 2009).

3 Data Collection

To obtain training data, we draw on the Cornell Legal Information Institute (Cornell LII)'s repository of US Supreme Court opinion texts.[3] We began by scraping all text associated with all opinions available through the Cornell LII site. For each case, we then removed all HTML markup, editorial information, and other non-opinion language (e.g. footnotes, case summaries, or front matter), and split the remaining text into paragraphs. For each paragraph, we then checked whether that paragraph contained a hyperlink to a section of the US Constitution. If any hyperlinks were present, we stored the paragraph and linked Constitution section(s) as training pair(s). Finally, we removed any duplicate paragraphs. This process left us with a total dataset of $\sim$ 328k unique paragraphs, of which $\sim$ 8k contained at least one link, and a total of $\sim$ 11k links.

Inspecting these data, we noticed that the annotation was inconsistent and incomplete. For example, not all paragraphs with the phrase "First Amendment" linked to the First Amendment. To solve this problem, we manually created a small list of rules for annotation. The list contained about 100 rules, and consisted mainly of mapping amendment names ("Seventh Amendment" or "7th Amendment") to the correct label. We also included several representative phrases, such as "free speech", "due process clause", and others. After this annotation, we had $\sim$ 36k paragraphs with at least one link, and $\sim$ 41k links total.

Though convenient, this process created certain trivial dependencies between linked paragraphs, which might limit a model's ability to generalize. Because hyperlinks and rules are associated with text, all linked paragraphs necessarily contain rule strings that correspond to the linked Constitution section. For example, all paragraphs that link to the First Amendment necessarily contain rule strings such as "First Amendment", "1st Amendment", or "Amendment I". A model trained on this dataset would likely treat the presence/absence of strings like these as strong classification signals, which is undesirable if the goal is to identify links between paragraphs that do not explicitly mention the name of the linked paragraph.

To encourage the model to move beyond these trivial patterns, we therefore create a modified copy of our training set, which we term the "stripped" dataset. In the "stripped" dataset, we randomly select half of the training examples, and delete all hyperlink or rule strings that occur within the text of these training examples, leaving potentially disfluent sentences. We delete hyperlink and rule strings from only half of training examples because presence of a phrase such as "First Amendment" is still a strong linking signal which we would like to preserve.

In our evaluations, we assess model performance on both the original and "stripped" datasets separately. We emphasize that this "stripping" process does *not* change the number of observations in either our training or evaluation sets. Instead, the "stripping" step simply removes rule strings from certain training examples, which (we suggest) compels our downstream tagging model to move beyond simply re-learning the rules we use to construct our training set.

[3] https://www.law.cornell.edu/

3.1 Manual Annotations

To assess model performance, we hand-annotated Constitutional references in all paragraphs ($n = 1241$) from an additional five Supreme Court cases: *Griswold v. Connecticut* (381 U.S. 479), *Miranda v. Arizona* (384 U.S. 436), *US v. Nixon* (418 U.S. 683), *Texas v. Johnson* (491 U.S. 397), and *NFIB v. Sebelius* (567 U.S. 519). We emphasize that these cases were *not* selected randomly. Since most Supreme Court cases infrequently reference the Constitution, we chose these cases because they litigate important constitutional law questions, and are therefore likely to contain a high density of positive examples with which to assess model performance.

These five cases provide two other desirable properties for an evaluation set. First, each of these cases addresses a different legal issue (e.g. criminal rights in *Miranda*; free speech in *Texas v. Johnson*). As a result, each case is likely to contain references to a distinct set of Constitutional provisions. Second, these cases also vary substantially in rhetorical style. For example, Justice Douglas's opinion in *Griswold* famously references and connects the "penumbras" of various Constitutional provisions in order to identify a right to privacy. Since most of these references consist of passing references to standard provision names (e.g. "First Amendment"; "Due Process Clause"), we expect the links in *Griswold* to be "easy" to predict. By contrast, *US v. Nixon* and *NFIB v. Sebelius* tackle abstract questions regarding the scope of Presidential and Congressional powers. As a result, they are more likely to indirectly reference Constitutional provisions, thus providing a more substantial challenge, and more room to improve.

4 Methods

Our data are defined in terms of input documents D, and reference documents C. The goal of the task is to link an input document $d \in D$ to zero or more reference documents. Formally, we will create a model of the form $h(d) \rightarrow \mathbf{y}$, where $\mathbf{y} \in \{0,1\}^{|C|}$ and $\mathbf{y}_i = 1$ if d is to be linked to document $c_i \in C$. If $\mathbf{y} = \{0\}^{|C|}$, this means that d links to no paragraph (true for most pieces of text, including this paragraph).

We emphasize that our aim in this preliminary work is not to discover the best architecture for this task, but to provide strong baselines for future work to build on.

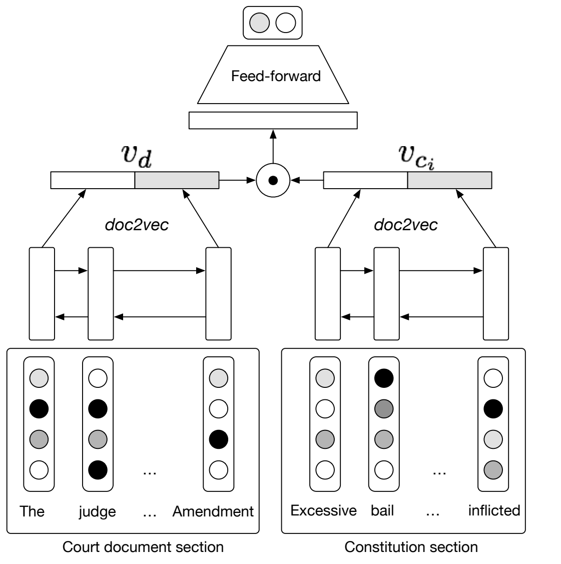

Figure 1: Diagram of the neural network architecture for a single binary classifier in the multilabel space. Token embeddings are from BERT, and we use a single *doc2vec* for both paragraphs. This system is described in Section 4.3, and corresponds to equations (1)-(3). The $\odot$ refers to element-wise multiplication.

4.1 Rule-based

An intuitive baseline is to use the rules defined for the annotation process as the entire labeling strategy. Instead of applying these rules to the training data, we apply them to the test data directly. As with any rule-based system, we would expect that this achieves high precision and low recall.

4.2 Linear Model

Beyond the rule-based system, we also implemented a linear multi-label classifier. Our implementation is a variant of the so-called Binary Relevance framework (Boutell et al., 2004; Nam et al., 2014), which builds a separate classifier for each label. As such, the problem decomposes to building C separate classifiers: $h(d) \rightarrow \{h_i(d)\}_{i=1}^{|C|}$, where $h_i(d) \rightarrow \{0,1\}$.

We used logistic regression as the model, and used unigrams, bigrams, and trigrams as features. Since the training data is wildly imbalanced towards unmatched paragraphs, we employ two tricks in our training. First, as a preprocessing step, we selected all examples with links, but subsampled the unmatched examples such that there was an equal number of matched and unmatched. Second, we downweighted all negative examples in training by a constant factor. This deals with the fact that every positive example for one class

Model	P	R	F1
Rule-based	**91.8**	47.0	62.2
Linear (original)	79.0	45.8	58.0
Linear (stripped)	68.3	54.3	60.5
Neural Network (original)	82.1	46.8	59.6
Neural Network (stripped)	76.5	**56.2**	**64.8**

Table 1: Results on the manually annotated test set. The top row uses the rule based classifier. The bottom two rows use the neural network model trained on the original and stripped training sets respectively.

is a negative example for all other classes, and also that the quality of annotation is unsure, as in (Liu et al., 2003).

4.3 Neural Network Model

In addition to the two prior baselines, we model this problem using a neural network classifier.

Inspired by work in dataless classification (Chang et al., 2008), a key observation in this model is that each element in the output vector **y** represents a *document*, not just a label. Under this observation, we can create a meaningful representation for each label which gives an additional signal for classification. As such, we use the index i to retrieve c_i, and rewrite the decision function as $h_i(d, c_i) \rightarrow \{0, 1\}$. Ultimately, we define a single model for all $h_i(\cdot)$ as follows:

$$v_d = d2v(T(d)),\ v_d \in \mathbb{R}^{2k} \quad (1)$$

$$v_{c_i} = d2v(T(c_i)),\ v_{c_i} \in \mathbb{R}^{2k} \quad (2)$$

$$h_i(d, c_i) = f(v_d \odot v_{c_i}) \quad (3)$$

Where T is a token embedding function, $d2v$ is a document embedding function (with hidden states of size k), and f is a feed forward neural network layer that projects to two dimensions. Loosely speaking, the function h_i can be understood as measuring the similarity between the vector representations of d and c_i. We used allennlp to build our systems (Gardner et al., 2017). Our architecture is depicted in Figure 1.

For the token embedding layer T, we used the BERT base cased pretrained embeddings (Devlin et al., 2018), as provided by huggingface.[4] For the document embedding layer $d2v$, we used a bidirectional LSTM with hidden size 300, 2 layers, and dropout 0.5. This embedder converts a sequence of embeddings into a fixed length by running the bidirectional LSTM over the sequence and concatenating the resulting context vectors from each direction. This document representation then has length equal to twice the hidden dimension of the LSTM, corresponding to the concatenation of the left and right context vectors.

We employed the same negative sampling and negative reweighting techniques for this model as described for the linear model.

4.4 Evaluation

During training, we tuned according to a split of the original train data. Since this data is automatically generated, it is not a good indicator of performance. Instead, we report all of our results on the manually annotated test set, described in Section 3.1. Since the decision from most of the classifiers will be 0, we evaluate the outputs of our model using F1 measure, calculated without regard to any individual class.

All of our code, data, and trained models are available online.[5]

5 Results and Analysis

Table 1 shows our main results. As expected, the rules-based approach gives high precision but low recall on the manually annotated set.

Interestingly, the linear and neural models trained on the original (unstripped) data achieve similar recall as the rule-based method, but suffer in precision. One explanation is that the imbalanced distribution of labels in the training set leads to overfitting of frequently-attested labels (hence the similar recall), and poor performance on all others (hence the drop in precision). The examples in Table 2 reinforce this idea.

Finally, the "stripped" results for each model show lower precision but higher recall relative to the original setting. We consider this an encouraging first step, which shows that the rule-stripping approach is important to prevent the model from simply re-learning deterministic training set rules. This pattern is particularly noticeable for the neural network model, which achieves the highest recall and highest overall performance of all approaches when trained on the "stripped" data.

In Table 2, we show some examples of pre-

[4] https://github.com/huggingface/pytorch-pretrained-BERT/

[5] https://github.com/mayhewsw/legal-linking

Input sentence	Rule Pred.	NN Pred.
The defendant argued that their right to **free speech** had been chilled.	First Amendment	First Amendment
This was a **Second Amendment** case.	Second Amendment	Second Amendment
This was a **Ninth Amendment** case.	Ninth Amendment	Sixth Amendment, Eighth Amendment
The court argued that the punishment was not only cruel, but also unusual.	Unmatched	Eighth Amendment
The decision in Escobedo v. Illinois, 378 U.S. 478, stressed the need for protective devices to make the process of police interrogation conform to the dictates of the privilege.	Unmatched	Fifth Amendment

Table 2: A comparison of predictions from the rule-based system and the neural network model (stripped). The linear model is omitted to save space. **Bold text** represents text matching a rule. The three table sections correspond to examples on which 1) both models are correct 2) only rule-based is correct, and 3) only the neural network is correct. The last example is taken from the manually annotated test examples, with some formatting removed.

dictions from the rule-based system and the neural network model (the linear model is omitted to save space). The table has three sections, corresponding to examples on which 1) both models are correct, 2) only rule-based is correct, and 3) only the neural network is correct. The example from the second section ("Ninth Amendment case") is interesting in how it contrasts with the nearly identical sentence above it ("Second Amendment case"). Naturally, the rule-based system correctly tags both, but the neural network is only correct on the "Second Amendment" sentence. This is likely because of imbalances in the training data, such that sentences with the phrase "Second Amendment" are common, but sentences with the phrase "Ninth Amendment" are much less common. In fact, in the training split we used, the phrase "Ninth Amendment" appeared less than 10 times out of nearly 40K examples.

The bottom section shows the power of the neural network model. Words such as "cruel", "punishment", and "unusual" are distinctive of the Eighth Amendment, even though they are in a different order. Similarly, "the privilege" is a common shorthand for the Fifth Amendment's protections against self-incrimination ("No person [...] shall be compelled in any criminal case to be a witness against himself" $\rightarrow$ "the privilege against self-incrimination" $\rightarrow$ "the privilege"). Such examples are of particular interest to legal practitioners, but are difficult to capture in a rules-based framework.

6 Conclusions

We have introduced a new task for linking portions of text from Supreme Court cases to the US Constitution, some data supporting this task (although with incomplete annotations), and some baseline models, including a rule-based system, a linear model, and a neural network system. Although the neural network system outperforms both the rule-based and linear systems, there is still further exploration to be done both in the direction of automatic or distant labeling, and in problem modelling. We look forward to other researchers using this dataset for future work.

From a practical perspective, we anticipate that this dataset could be used to give valuable insights on research questions of interest to the world of political science. For example, these data could be used to study which amendments tend to see higher litigation rates according to the period in the Supreme Court, or rhetorical co-citation of Constitution sections.

Acknowledgments

This work was supported in part by a gift from Google and by contract HR0011-15-2-0025 with the US Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. 2004. Learning multi-label scene classification. *Pattern recognition*, 37(9):1757–1771.

L Karl Branting. 2017. Data-centric and logic-based models for automated legal problem solving. *Artificial Intelligence and Law*, 25(1):5–27.

Pompeu Casanovas, Monica Palmirani, Silvio Peroni, Tom van Engers, and Fabio Vitali. 2016. Semantic web for the legal domain: the next step. *Semantic Web*, 7(3):213–227.

Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *AAAI*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Daniel Duma and Ewan Klein. 2014. Citation resolution: A method for evaluating context-based citation recommendation systems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 358–363.

Zachary Elkins, Tom Ginsburg, James Melton, Robert Shaffer, Juan F Sequeda, and Daniel P Miranker. 2014. Constitute: The worlds constitutions to read, search, and compare. *Journal of Web Semantics*, 27:10–18.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 179–186. IEEE.

Robert G McCloskey and Sanford Levinson. 2016. *The American supreme court*. University of Chicago Press.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classificationrevisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer.

Tadashi Nomoto. 2018. Resolving citation links with neural networks. *Frontiers in Research Metrics and Analytics*, 3:31.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics.

T Schwartz, M Berger, and J Hernandez. 2015. A legal citation recommendation engine using topic modeling and semantic similarity. In *Law and big data workshop, 15th international conference on artificial intelligence and law*.

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.