Natural Language Processing and Machine Learning for Law and Policy Texts

John Nay[1]

NYU

April 7, 2018

1. Introduction

Almost all law is expressed in natural language; therefore, natural language processing (NLP) is a key component of understanding and predicting law at scale. NLP converts unstructured text into a formal representation that computers can understand and analyze. The intersection of NLP and law is poised for innovation because there are (i.) a growing number of repositories of digitized machine-readable legal text data, (ii.) advances in NLP methods driven by algorithmic and hardware improvements, and (iii.) the potential to improve the effectiveness of legal services due to inefficiencies in its current practice.

NLP is a large field and like many research areas related to computer science, it is rapidly evolving. Within NLP, this Paper focuses primarily on statistical machine learning techniques because they demonstrate significant promise for advancing text informatics systems and will likely be relevant in the foreseeable future.

First, we provide a brief overview of the different types of legal texts and the different types of machine learning methods to process those texts. We introduce the core idea of representing words and documents as numbers. Then we describe NLP tools for leveraging legal text data to accomplish tasks. Along the way, we define *important NLP terms* in italics and offer examples to illustrate the utility of these tools. We describe methods for automatically summarizing content

[1] Email: jnay@nyu.edu

(sentiment analyses, text summaries, topic models, extracting attributes and relations, document relevance scoring), predicting outcomes, and answering questions.

2. Legal Text

We divide legal texts into five primary types: constitutional, statutory, case, administrative, and contractual. The legislative branch of the government creates statutory law; the executive branch creates administrative rules; the judicial branch creates case law in the form of court case opinions; and private parties create contracts. Laws are found at varying levels of government in the United States: federal, state and local.

Adopted versions of public law are often compiled in official bulk data repositories that offer machine-readable formats. Statutory law is integrated into the United States Code (or a state's Code if state-level), which organizes the text of all Public Laws that are still in force into subjects. Administrative policies become part of the Code of Federal Regulations (or a state's Code of Regulations), which is also organized by subject. Case law is created by judges writing opinions for rulings on court cases. Examples in this Paper leverage bulk data repositories of law to train and test machine learning NLP models.

Different types of law possess characteristics that make certain computational methods more relevant. The more uniform the layout and the more predictable the content, the more amenable the documents are to automating their conversion to formal representations. There can be large variation in the content of judicial opinions due to their focus on the concrete facts of real-world cases. Administrative regulations implement legislation and are thus usually more specific and detailed than the corresponding statutes. Overall, public laws and regulations follow regular patterns

and their content is relatively structured. However, public law does not attempt to cover all the contingencies and possible scenarios that may occur, and statutes must often be interpreted by judges or regulators. On the other hand, private contracts attempt to cover a large number of possible outcomes relevant to the relationship being formalized. This suggests that contracts have a higher chance of moving out of the messy ambiguous world of natural language than public law and, indeed, the rise of distributed-ledgers and related information technologies may be accelerating this shift.

3. Machine Learning and NLP

Machine learning is the process of training a computational model to accomplish a task with data. There are two primary task categories: prediction and data exploration/description. Prediction is accomplished by a subset of machine learning called *supervised learning* where *observations* (e.g. Congressional bills) composed of pairs of (i.) predictor variables (a bill sponsor's political party) and (ii.) outcome variable (whether the bill was enacted into law) are used to learn a model that can take in a new observation's measurements of the same predictor variables (a new bill's sponsor party) and predict its outcome (enactment). If the outcome predicted is a real-valued number, e.g. number of votes for a bill, then the model is called a *regression model*. If the outcome predicted is a category, e.g. enacted or failed, the model is called a *classification model*. The model learning process involves making predictions with the model, measuring the prediction error, adjusting the tunable parameters of the model to reduce prediction error on that training data, and repeating this process until the parameter adjustments suggested by the learning algorithm are negligibly small, i.e. the model has *converged*. The primary goal of supervised learning is to learn a model that will generalize from the sample of data

that it was trained on, the *training data*, to new data, *testing data*. The model can then be used in real-world situations where the outcome is unknown, but the predictor variables are known, to forecast the value of the outcome.

Data exploration and information retrieval tasks are facilitated by techniques called *unsupervised learning*. In contrast to supervised learning, in unsupervised learning, observations only include their measured variables and no particular variable has the special status of the *outcome variable* to be predicted. This greatly increases the quantity of data available because most data are not explicitly labeled with an outcome of interest. For example, there is a vast amount of raw text data on the internet. A legal informatics task suited for unsupervised learning is finding other Congressional bills that have similar policy content to a given Congressional bill.

Measuring a model's performance on supervised tasks is usually more straightforward because there are standard measures of predictive performance, e.g. accuracy for a classification task or mean-squared error for a regression task. On the other hand, the tasks of automatically finding similar documents or clustering together documents into coherent groups usually have no objectively correct answers available. Even expert human judges can disagree on the results. Therefore, validating unsupervised learning systems is more difficult than validating supervised learning systems.

An important component of a workflow for both types of machine learning is converting raw, unstructured data into a suitable computational representation. The first step of many machine learning approaches to NLP is creating a numeric representation of text. There are two primary methods used to represent words and documents computationally and both usually require that we have a *dictionary* of the vocabulary words that exist in the *corpus* (collection of texts). This dictionary is a list of words (or, less often, characters) to take into account in the analysis.

The *one-hot-encoding* method represents a collection of words, e.g. a sentence, as a list of 1s and 0s that is the same length as the number of words in the vocabulary. There is a 1 if the word represented by that location in the list appears in the sentence and 0 if that word does not appear. For each word, there is an *indicator variable* denoting whether the word occurs. A sentence is represented as a long list of 0's and a few 1's, which is called a *sparse representation*. Instead of indicating only the presence of a term in a document we can count the number of times the word occurs. This *term frequency* representation, is generally more effective for document retrieval tasks, while term presence is more effective for sentiment analysis (Pang et al. 2002; Pang and Lee 2008). The *continuous-space* method represents each word with a dense vector of real-valued numbers, e.g. 'textbook' = [0.02, 0.3, -0.1]. The values of these numbers are learned from data (one process for learning them, *word2vec*, is described below).

For most machine learning models, observations need to be represented by the same set of variables. Therefore, if we are modeling phrases, sentences, and documents, we need some way to convert the varying-length strings of words into a fixed number of variables. One approach is to treat a document as a *bag-of-words* and effectively ignore the word order.[2] With one-hot-encodings, a bag-of-words representation of a document is a list of which words are in the document and how often they each occur, but no information about where the words are located within the document. This is a sparse representation because there are primarily 0's. With *dense representations*, the (equal-length) vectors representing each word in the document can be averaged, summed or otherwise combined to obtain a single representation for the document as a whole.

Whatever specific numeric representation technique we use, the same process for obtaining a single representation of a document can be applied to all (variable length) documents in an analysis

[2] The word order is ignored when we count only single words (called unigrams), but when we use n-grams where n > 1 the local order of the words is at least partially captured.

so the final machine learning system has equal-sized representations. This allows us to apply further numerical processing techniques to the resulting vectors, including an array of machine learning models. For instance, if we obtain vector representations of a collection of texts we can apply clustering algorithms directly to these representations to automatically group similar documents together to facilitate searching through a large corpus (Fig. 1). Or we can apply supervised learning models that predict an outcome related to the text. The possibilities are almost endless. We now describe the NLP tasks that are most relevant to legal informatics.

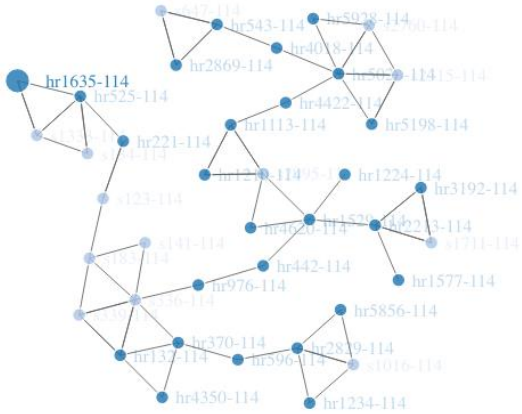


Fig 1: Example of a network of similar bills under consideration by Congress. House bills are dark blue and Senate bills are light blue.

4. NLP Tasks and Tools

We divide our discussion of NLP tasks and tools into: summarizing content, extracting content, retrieving documents, predicting outcomes correlated with text, and answering questions.

4.1. Summarizing Content

There are large amounts of text related to law and policy. For instance, we may be interested in determining how the public reacts to the announcement of a Supreme Court decision on social media. There are too many social media posts to read all the content, but we would like a summary to understand the general reaction to the case outcome. In this sub-section, we discuss methods for estimating the emotion expressed within text, creating short summaries of longer texts, and discovering the main themes and topics of a corpus. These tools can augment human synthesis abilities and partially automate the process of quickly obtaining insight across large collections of texts.

4.1.1. Sentiment Analysis

Sentiment analysis tools attempt to automatically label the subjective emotions or viewpoints expressed by text. For instance, "I really enjoyed the paper" expresses positive sentiment, whereas "there were terrible example phrases in the paper" is negative. A practical application to law and government is obtaining public comments on rules and regulations and then scoring them with sentiment analysis models to provide regulators with the public reaction to pending policy (Cardie et al 2006; Kwon et al 2006).

Sentiment labels may be on a numeric scale from positive to negative (*sentiment polarity*) or more specific emotion classes such as excitement or pride. If on a numeric scale and at the word-level, the scores for all the words in a document can be averaged to assign an overall score for the

document.[3] If categorical emotion labels, the proportion of words assigned to each emotion can summarize the sentiment. Most sentiment analysis techniques can be divided into either dictionary-based or learning-based. Dictionary-based techniques have a large list of words that previously have been manually scored/rated for their subjective sentiment.[4] This scoring is usually at the individual word-level (*unigrams*) because sets of more than one word (*bigrams, trigrams*, *etc.*) occur much less often and therefore scoring their sentiment would have less practical value in automatically labeling new sentences. *Dictionary-based methods* work relatively well for tracking public sentiment on Twitter (Dodds et al. 2011). However, they can perform poorly where negation and sarcasm are present. Take, for instance, the sentence "he is not happy or loved." We input this sentence into a simple dictionary-based emotion detection algorithm (Mohammad and Turney 2010), which described the sentence as 1/3rd anticipation, 1/3rd joy, and 1/3rd trust. We also input the sentence into a dictionary-based numeric sentiment algorithm (Nielsen 2011), which scored it as moderately positive.

To improve a dictionary-based system, sentences can be automatically parsed and the function that a word serves within the sentence can be labelled with separate tools (see below for information on parsing), then rules of negation, and more generally, *valence shifters*, can be added into the computation of sentiment (Kennedy and Inkpen 2006). Dictionary-based approaches may not generalize well to texts in a specialized domain where important words are too rare to be previously scored in a dictionary or where words that are scored serve different purposes in different contexts. This is important for legal text because most existing sentiment dictionaries are scored by non-legal experts and in the context of general texts such as news articles.

Machine learning-based sentiment analysis methods can leverage the data within a corpus to build a model that predicts sentiment based on the string of words. For example, this approach has

[3] However, the sentiment is often bi-modal and in these cases the average can be misleading (Pang and Lee 2008; Carenini et al. 2013).
[4] This is often accomplished by paying people to rate the words.

been applied to predict law-makers' support or opposition to policy issues from their Congressional floor-debate transcripts (Thomas et al. 2006). The machine learning model will often take an entire sentence or document as input and predict the sentiment for the whole collection of words. The user must (i.) label the sentiment of a sufficient number of documents manually, (ii.) use one of the methods described above to map the texts into numeric representations, and (iii.) learn a prediction model that outputs a sentiment label for any given text representation input. This prediction model would be trained on the labeled documents and then could be deployed to predict unlabeled, unseen documents.

A drawback to this approach, similar to dictionary-based methods, is that if a machine learning model is trained to predict sentiment within one domain it may not transfer well to a different domain (Owsley et al. 2006; Reed 2005) and most documents with sentiment labels are not legal documents. Context matters for subjective sentiment. For example, the phrase "go read the book" represents positive sentiment in a book review and negative sentiment in a movie review (Pang and Lee 2008).

4.1.2. Textual Summaries

This group of tools includes methods that automatically convert longer texts into shorter texts. This includes converting longer documents into shorter documents or converting entire corpora into some type of informative summary of all the documents. There are two general approaches to this task. The simpler approach, *extractive summarization*, identifies important portions of a text (words, phrases or full sentences), extracts them, and combines them into a summary. The more difficult, but potentially more powerful approach, *abstractive summarization*, involves generating entirely new text that was not necessarily found in the text that is being summarized. This requires

the algorithm to build a complex representation of the text that captures its essence, condition on that representation, and generate a grammatically correct smaller block of text that expresses the essence of the larger block.

At this point in time, for texts longer than a paragraph or two, purely abstractive-based summarization techniques are outperformed by extractive summarization. A widely applied extractive techniques is TextRank (Mihalcea and Tarau 2004). TextRank creates a graph structure where each vertex is a sentence in a document; determines the similarities between the sentences based on the number of words sentences share, normalized by their lengths; uses these similarity relations as edges between the graph vertices; applies a graph-based ranking algorithm to score the importance of the sentences; and finally includes the most important sentences in a summary. The most famous *graph ranking algorithm* is Google's PageRank, which uses links between webpages to form a large graph. Graph ranking algorithms work by determining the number of recommendations for a given vertex and the value of the recommendation. A vertex with many highly-valued recommendations is deemed important. In the webpage example, page X is recommended by page Y if Y links to X, and the importance of that recommendation depends on how many pages recommend (link to) Y, which is recursively computed by running the algorithm repeatedly until importance scores are no longer changing much. In the text example, a sentence recommends another sentence if it has similar text.

4.1.3. Topic Models

An algorithm like TextRank can often provide high quality summaries of longer texts but it is designed for summarizing one document at a time. If we have a large collection of documents on a potentially wide range of topics, then an overview of the various topics and how much each

document is devoted to each topic may provide a more useful summary. A *topic model* is a mixed-membership probabilistic model of distributions over words for a corpus (Blei et al. 2003). A topic is just a list of words, e.g. an environmental topic may be described by "recycle, planet, clean, environment."

The topic modeling algorithm can be described by a generative process: create topics for an entire corpus, choose a topic distribution for each document, then for each word in each document: choose a topic from that document-level distribution of topics and then choose a vocabulary term from the topic, which is a distribution over the terms in that corpus. This models documents as being composed of multiple topics to varying degrees. For a given number of topics, estimating the parameters of the model automatically uncovers the topics spanning the corpus, per-document topic distributions, and per-document per-word topic assignments (Blei 2012). A correlated topic model explicitly represents variability among topic proportions, allowing topical prevalence within documents to exhibit correlation (Blei and Lafferty 2007), e.g. a climate change topic can be more likely to co-occur in a Judicial opinion with a high proportion of words from an energy topic than in an opinion with a high proportion of words from a financial regulation topic (see Fig. 1 for an example from Presidential texts).

We often have important *data about text data*, which is called *metadata*. The topic model has been extended to incorporate text metadata on time, location and author (Blei and Lafferty 2006; Rosen-Zvi et al. 2010; Eisenstein et al. 2010), and integrated with models of legislative voting from political science (Gerrish and Blei 2012). The *structural topic model* (Roberts et al. 2013) extends the correlated topic model by modeling topic prevalence, the proportion of a document devoted to a topic, as a function of the document-level variables. This allows us to flexibly model the relationship between document characteristics and topic prevalence. The distribution over words (the actual content of the topics) is also adapted so that it is modeled as being affected by a combination of

topics, document metadata, and interactions between topics and the metadata. In this way, both the *prevalence* and the word *content* of topics can be modeled as a function of document metadata, allowing us to test hypotheses about the effects of any metadata, e.g. the author of a document, on the topics expressed.

We demonstrate the power of the topic modeling approach with an example. There is controversy over the U.S. President creating law and policy through actions such as Executive Orders, but there is little rigorous research on the topic that leverages the full texts of Presidential actions. Ruhl, Nay and Gilligan (2018) applied the structural topic modeling approach to all Presidential direct action documents to understand what policy topics are shifting to/from a statutory requirement type of Presidential action (Proclamations or Determination) to/from a more unilateral type of Presidential action (Memorandum or Executive Order). We also analyzed whether the language used to describe these topics was changing over time from President to President.

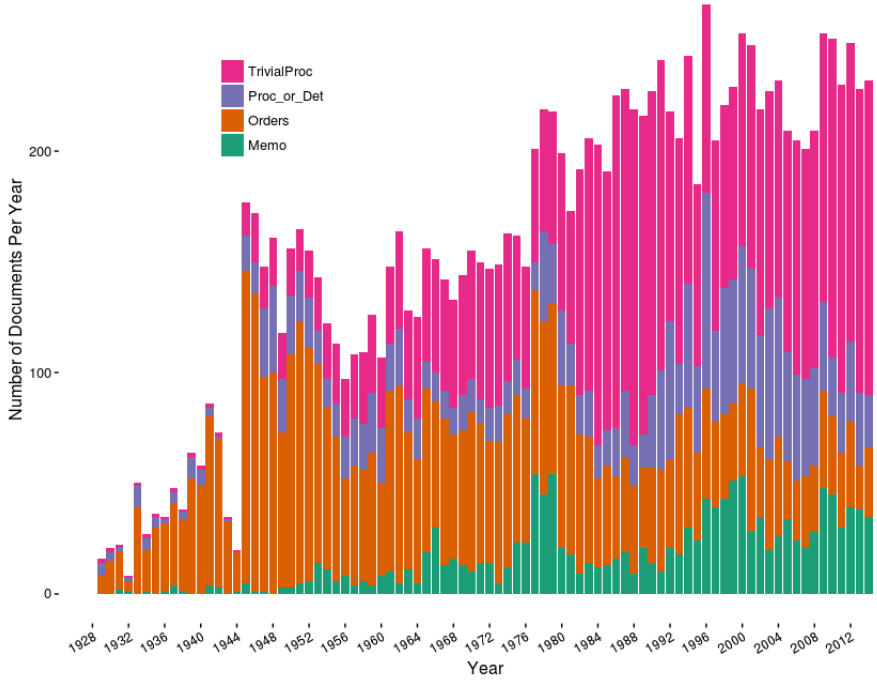
We created a text corpus consisting of all documents through which unilateral presidential law-making authority has been exercised[5] and analyzed the documents from early 1929 through 2015 to model how policy topics change during this time and how that interacts with the type of Presidential action.[6]

A first step of most NLP tasks is *tokenization* of the text. A tokenizer divides a document into its individual words. This can often be accomplished by simply separating words by the whitespace between them. After tokenization of this Presidential text, we converted all letters to lower-case and removed numbers, punctuation, and *stop words*, common words that would be found across topics and documents and therefore add little value in creating distinct topics, e.g. "the". We also *stemmed*

[5] We scraped all Presidential Memorandum (1,465), Presidential Determinations (801), Executive Orders (5,634), and Presidential Proclamations (7,544) available on Gerhard Peters and John T. Woolley's *The American Presidency Project* (www.presidency.ucsb.edu), which is the most comprehensive collection of Presidential documents.

[6] See Ruhl, Nay and Gilligan (2018) for a much more comprehensive analysis.

words with the Porter stemmer (Porter 1980). A stemmer removes the endings of many words, e.g. consolidate, consolidated, and consolidating would all be converted to "consolid." These are common *pre-processing* techniques applied to text data before unsupervised modeling to reduce the dimensionality and complexity of our text representation in a way that attempts to capture the most important parts of the words and overall document.[7] As a final pre-processing step, we converted each document to a one-hot-encoded bag-of-words representation, an integer vector of frequencies of terms occurring in at least five documents. By using only terms occurring in at least five documents, this removed 16,864 of 25,653 terms. Our final corpus had 13,730 documents, 8,789 terms and 1,563,608 tokens (individual words).



[7] For unsupervised tasks, it is usually advisable to apply stemming and removal of stop-words; however, if there are a sufficient number of training observations, these techniques can actually degrade the performance of a supervised learning model (Manning et al. 2008).

Fig. 2: Number of Presidential Proclamations or Determinations, trivial Proclamations, Memorandums, and Executive Orders from 1928 through 2015.

Each document is either a Presidential Memorandum, Presidential Determinations, Executive Orders or Presidential Proclamation. Most proclamations with substantive weight are authorized by statute requiring Presidential issuance of the Proclamation to trigger policy programs, e.g. disaster aid; whereas Orders spring from unilateral Presidential action. Memorandums, while less attention-grabbing, are legally similar to Orders. Therefore, if we observe a shift in a topic from Orders to Memorandums, this may suggest that the President is trying to downplay exercise of power for that topic. Because many of the Proclamations are trivial we created a cateogory for trivial proclamations ("TrivialProc"), which is any Presidential Proclamation that has at least one of these terms in the title: "Day", "Week", "Month", "Anniversary." Because the non-trivial Proclamations and Determinations are similar from a legal perspective we grouped them into a category.

We estimated a 50-topic model and the effect of the year, the type of Presidential action, and their interaction, on the expected proportion of a document that belongs to a given topic. To illustrate the results, Fig. 2 visualizes an environmental topic. Orders have over time been less likely to talk about the environment while Trivial Proclamations have been more likely. Table 1 illustrates the way in which the content of the topic varies with the President.

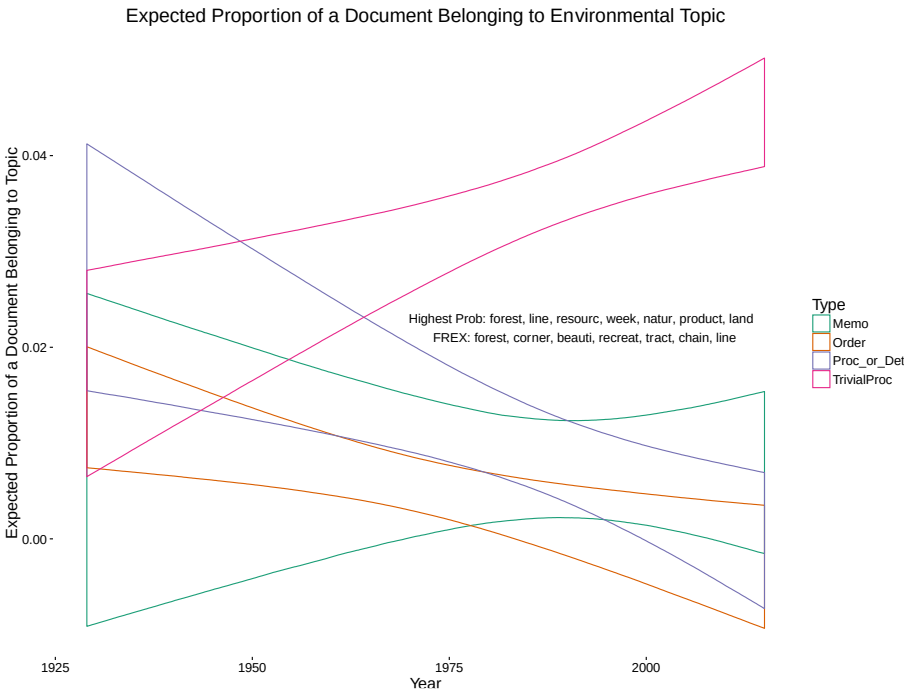


Fig. 3: The lines are the expected proportion of a document that belongs to the topic as a function of the year and document type, and the shading represents 95% confidence intervals. FREX words are words that are both frequent within a topic and exclusive to that topic compared to other topics (Airoldi and Bischof 2015).

President	**Environmental Topic Top Words**
Herbert Hoover	counti, specimen, coot, avenu, woodcock, gallinul, wood
Franklin D. Roosevelt	refug, bird, migratori, waterfowl, chain, lake, wild
Harry S. Truman	chs, center, meander, stone, line, intersect, corner
Dwight D. Eisenhower	rout, straight, junction, northeast, corner, easter, southwest
John F. Kennedy	recreat, outdoor, chain, timber, forest, urban, utah
Lyndon B. Johnson	canyon, beauti, rim, chain, norther, intersect, warrant

Richard Nixon	environ, forest, earth, beauti, wood, clean, hunt
Gerald R. Ford	forest, boat, forestri, clark, hunter, natur, resourc
Jimmy Carter	forest, trail, messag, wildlif, scenic, environ, environment
Ronald Reagan	food, forest, farmer, farm, hunger, abund, anim
George Bush	tree, hunger, forest, plant, beauti, arbor, rice
William J. Clinton	forest, abund, rural, anim, stewardship, beauti, sustain
George W. Bush	outdoor, forest, beauti, recreat, natur, wildlif, enjoy
Barack Obama	recycl, farmer, forest, planet, farm, clean, environ

Table 1: This topic captures environmental policy and shows how the words Presidents use to express their environmental policy change across individual Presidents.

Because we explicitly modeled the between-topic correlation within documents (Blei and Lafferty 2007) we are able to discover which topics are likely to occur in the same Presidential document as the environmental topic (Fig. 3).

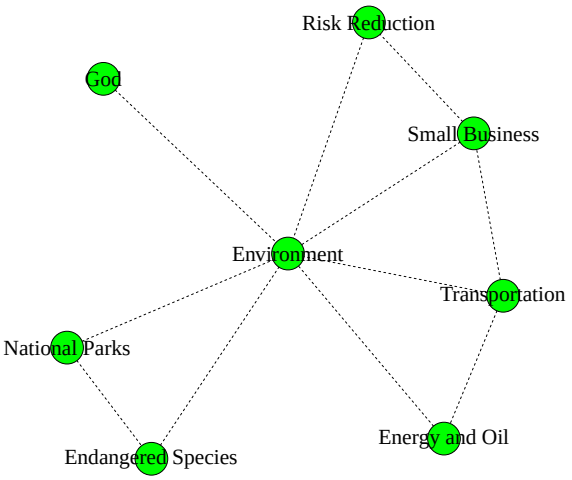


Fig. 4: Positive correlations (represented as lines) between topics indicate that both topics are likely to be discussed within a given Presidential document. This is the subset (of the larger graph of all 50 topics) that is correlated with the environmental topic.

4.2. Extracting Content

Content extraction is similar to summarization; however, sentiment analyses, text summaries, and topic models seek to obtain a holistic view of a corpus. With content extraction, the primary goal is to convert a large amount of text into a formal representation of specific fine-grained facts found in the texts. Content extraction is often a lower-level operation within a larger NLP pipeline that uses the extraction output as a first step. For instance, we may want to automatically extract all pairs of (i) dollar amounts and (ii) the description of the expense associated with the dollar amount from complex corporate contracts, and then use this information to compare thousands of contracts to determine what particular language surrounding the expenses led more or less litigation involving the contract. This analysis could inform the drafting of future contracts. Or, for example, we may want to find all the federal agencies mentioned in judicial opinions for the 2nd Circuit in the past ten years.

The simplest type of extraction is *attribute extraction* where we attempt to extract certain pre-specified attributes from a string of text (Russell and Norvig 2009), e.g. all dollar amounts in a judicial opinion. *Relational extraction* attempts to extract useful relationships among attributes (Russell and Norvig 2009), e.g. after determining a dollar amount, the system would determine the object the dollar amount is referencing. To demonstrate the output of these techniques, I applied the Stanford

Core NLP software toolkit (Manning et al. 2014) to the following sentence from a Tennessee state bill.[8]

> *Title 68, Chapter 201, Part 1, is amended by adding the following language as a new section: (a) As used in this section: (1) 'Covered electric-generating unit' means an existing fossil-fuel-fired electric-generating unit located within this state that is subject to regulation under EPA emission guidelines.*

The text was tokenized, the sequence of tokens was split into discrete sentences, the part-of-speech for every token was predicted, and whether each token is a named entities. The software also identified the syntactic relationships of the words in the sentences and determined if entity mentions throughout a document are referencing the same entity (co-reference resolution).

Part-of-speech tagging systems use manually labelled data to train supervised learning models to predict the part-of-speech of a word given the surrounding words (Toutanova et al. 2003).[9] Part-of-speech (POS) categories can include singular proper nouns (NNP), cardinal numbers (CD), third-person singular present verbs (VBZ), past participle verbs (VBN), prepositions (IN), adjectives (JJ), and more. Fig 4. demonstrates POS-tagging for the legislative text.

[8] The visualizations were created with the brat (http://brat.nlplab.org) visualization tool.
[9] Many state-of-the-art models for predicting syntactic and semantic characteristics of sentences explicitly represent the ordering of the words in a sentence: either by representing the one-dimensional structure of the flow of a sentence as it would be read by a human or the complex tree-like structure of relations between words. This is accomplished with complex machine learning models such as recursive and recurrent neural networks and conditional random fields.

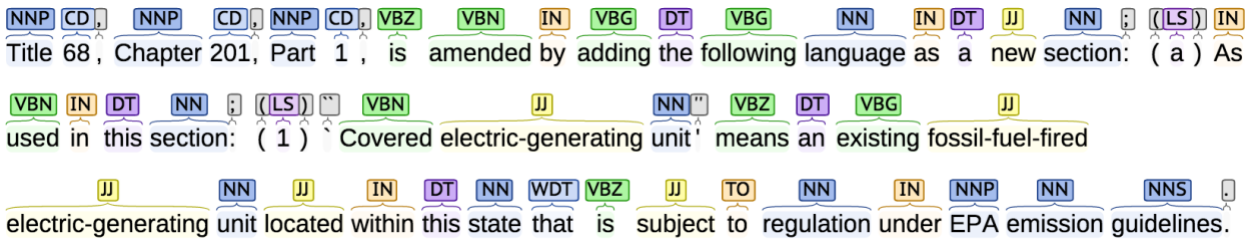


Fig. 5: Part-of-speech tagging for a sentence in a state bill.

Named-entity recognition predicts whether a token is a named (person, location, or organization) or numerical (money, date, time, duration or set) entity. Named entities are often predicted using supervised learning models trained on texts with words manually categorized into these classes (Finkel et al. 2005) and numeric entities are often predicted using simple hand-coded rules. The Environmental Protection Agency (EPA) was identified as an organization in the last part of the bill sentence (Fig. 5).

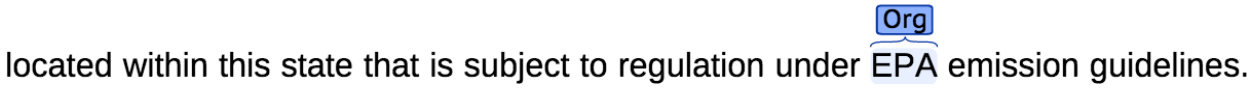


Fig. 6: Named entity recognition for the last part of the bill sentence.

Syntactic parsing systems predict the functional relationships between words in a sentence. This is a difficult task because of the inherent ambiguity of natural language. Supervised learning models are trained on large labeled corpora. The state-of-the art systems use neural networks (Andor et al. 2016). With long sentences, such as the legislative sentence in Fig. 6, there can be many syntactic relationships.

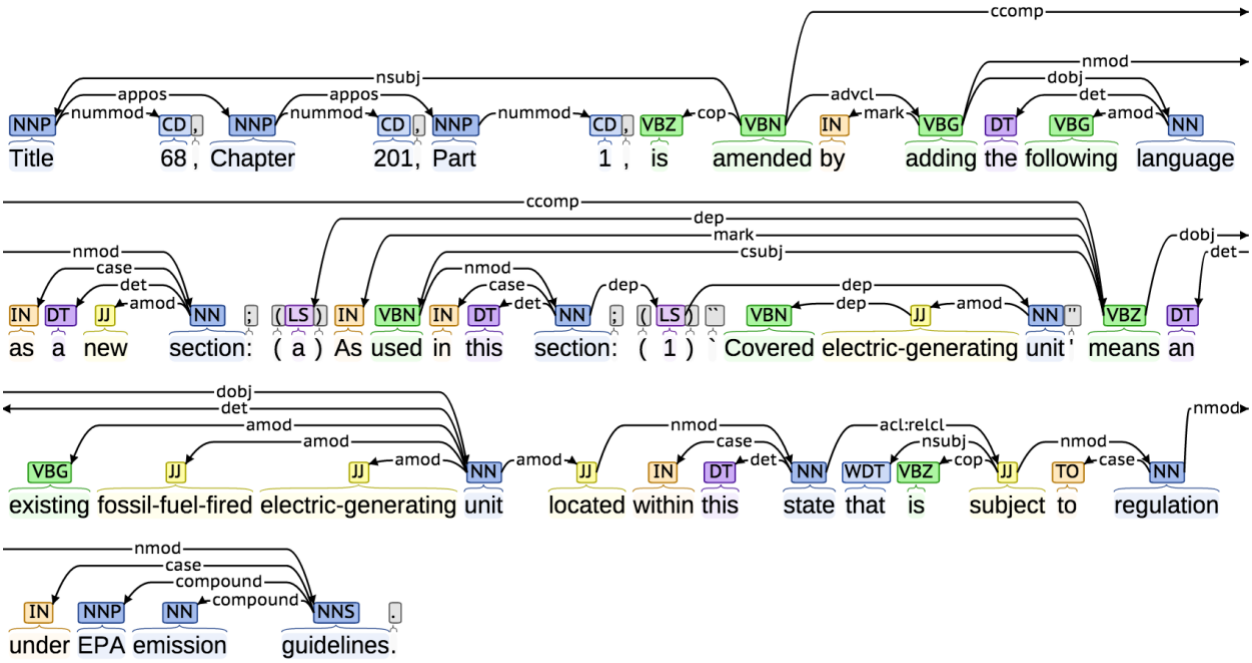


Fig. 7: Syntactic dependencies.

We also applied the Stanford Core NLP software to a larger section of a Tennessee bill to demonstrate *co-reference resolution* (Lee et al. 2013). This is best explained by an example. In Fig. 7, the mention of Subsection f was linked to a previous mention, and within the sentence the physical and internet-based notices are predicted to be referring to the same underlying concept of notice.

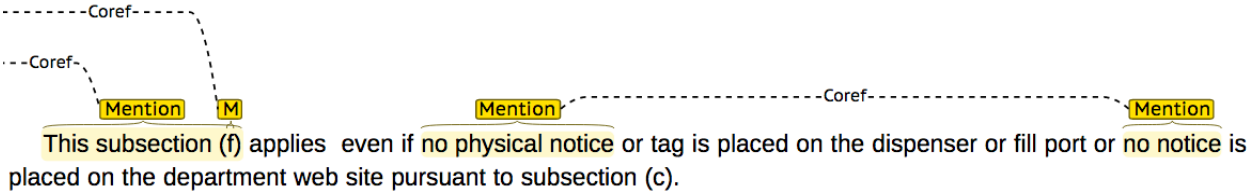


Fig. 8: Co-reference resolution.

4.3. Retrieving Information and Documents

Information retrieval (IR) tasks are characterized by a user's query, a set of documents to search, and the subset of the documents returned by the system (Russell and Norvig 2009). The results of a query over a set of documents is a list of documents that are relevant to that query. The simplest systems, *Boolean retrieval*, return documents that simply contain the words found in the query. More complex systems use machine learning. An advantage of using machine learning representations over Boolean search is that the machine learning approach provides a ranking over the documents based on how relevant they are to the query, whereas with Boolean search the system returns all documents that contain the terms in the query (Manning et al. 2008). If the machine learned vector representations capture the meaning of the documents and the queries, then this can outperform Boolean retrieval and provide the most relevant documents, and in the ideal ordering.

There are two primary measures of the performance of IR systems. *Precision* is the proportion of the returned documents that are relevant to the user's needs and *recall* is the proportion of all the relevant documents in the system that were returned to the user. When the corpus is very large and multiple documents may serve a similar purpose for a user, e.g. when searching for law review articles on administrative law, we usually are more interested in optimizing our precision, but in situations where it is important to ensure comprehensive coverage of a search query, e.g. in reviewing emails deemed as potentially relevant to a court case for the "smoking gun" piece of evidence, we are more interested in recall.

When we *cannot* expect the user to manually classify documents as relevant or not to their query, we can utilize unsupervised learning techniques and map the queries and the documents into a shared mathematical space to return documents located near the query within this space. When the user *can* interact with the machine learning system and provide feedback on the relevance of the

documents returned in an iterative process, then this information can be leveraged for supervised learning models that are tailored to a particular query or set of queries. This interactive approach with iterative human-computer review requires much more human input but is warranted when the stakes are high, e.g. during document review for a court case.

4.3.1. Unsupervised Learning

First, we describe unsupervised learning approaches. One of the simpler transformations from raw text into a mathematical representation is obtained by the *term-frequency inverse document frequency (tf-idf)* technique. The term frequency, *tf*, is how often the term occurs in a document, and the inverse document frequency, *idf*, is the logarithm of the total number of documents divided by the number of documents that contain the term. It is important to consider the *idf* because some words, such as "Section" in legislation, will occur very often across all documents in the collection and thus add little or no value in discriminating between documents. The *tf-idf* is computed by multiplying the term-frequency inverse document frequency and is therefore high when a term occurs very often in very few documents. In this way, *tf-idf* allows us to map documents and queries into vectors that are useful for discriminating between documents (Manning et al. 2008). These vectors are a bag-of-words representation because the order of the words is discarded. This can be problematic for subtle textual differences, e.g. "reversed the lower Court" and "the lower Court reversed" are represented by the exact same vector in a bag-of-words model but they can have very different legal implications within a judicial opinion.

Continuous-space vector representations of words can capture subtle semantics across the dimensions of the vector and potentially learn more useful representations of texts. One method to learn these representations, is to use a neural network model to predict a target word with the mean

(or sum or another transformation) of the representations of the surrounding words (e.g. vectors for the two words on either side of the target word in Fig. 8). The prediction errors are then used to update the representations in the direction of higher probability of observing the target word (Mikolov et al. 2013; Bengio et al. 2003). After randomly initializing representations and iterating this process, called *word2vec,* over many word pairings, words with similar meanings are eventually located in similar locations in vector space as a by-product of the prediction task (Mikolov et al. 2013). Using continuous-space vector representations of words and documents in an IR system may allow a query to return a document that is deemed similar and relevant to the query but not actually contain any words that were in the query (Nay 2016). One of the simplest methods of obtaining a continuous-space representation of a document is to average all the word vectors in the document that were learned with *word2vec.* This can create a representation that captures the general meaning of the entire document.

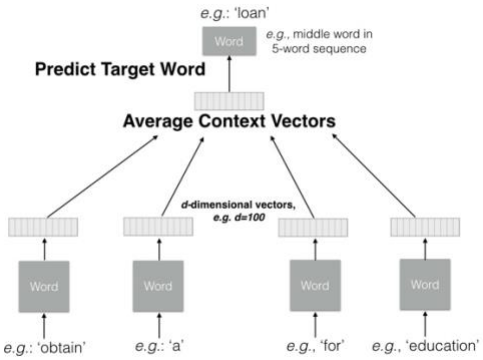


Fig 9: Word2Vec algorithm (Mikolov et al. 2013).

4.3.2. Supervised Learning

When the stakes are high, it can be worth spending the time to manually label documents. During the legal discovery process, there are often hundreds or thousands of documents to review for relevance. The electronic discovery (*e-discovery*) process of finding documents relevant to a case can also be viewed as a supervised learning problem where the user iteratively: provides relevant documents, has the machine use those to train a model to predict what other documents may be relevant, and from those candidates chooses more relevant documents.

The document review process is an important IR use-case. Federal judges have issued opinions permitting (*Da Silva Moore v. Publicis Groupe, 2012 U.S. Dist. LEXIS 23350 (SDNY, Feb. 24 2012)*, and sometimes requiring (*EORHB, Inc. v. HOA Holdings, C.A. No. 7409-VCL (Del. Ch. Oct. 15, 2012))*, the use of technology to assist lawyers in searching documents for evidence. There are often hundreds or thousands of documents that may be relevant to a court case and lawyers must find the relevant documents within this set. Reviewing every document manually is (i.) less efficient than a technology-assisted approach because many documents can easily be ruled out and should not be reviewed, and (ii.) potentially less effective because humans are error-prone (Roitblat et al. 2010).

E-discovery and technology assisted review (*TAR*) use supervised learning to improve the discovery process. An effective TAR technique for discovering nearly all relevant documents is continuous active learning (*CAL*) (Cormack and Grossman 2014). CAL consists of four primary steps: 1. Find at least one example of a relevant document; 2. Train a supervised learning model to predict relevance to the case using the document(s) from step 1 and then use the model to score the remaining documents and return the documents scored as most likely to be relevant; 3. Review the documents from step 2 and manually classify each as relevant or not; 4. Repeat the previous two steps until no suggested review documents are considered relevant (Grossman and Cormack 2016).

4.4. Predicting Outcomes

If an event of interest is correlated with text data, we can learn models of text that predict the event outcome. For instance, Kogan et al. (2009) predict financial risk with regression models using the text of company financial disclosures. Topic models have been used for predicting outcomes as a function of the proportions of a document that are devoted to the automatically discovered topics (Mcauliffe and Blei 2008). This has been applied to develop a topic model that forecasts roll-call votes from the text of Congressional bills (Gerrish and Blei 2011). An advantage to this prediction approach is that the model learns interpretable topics and the relationships between the learned topics and outcomes. A disadvantage of the topic model approach is that other (less interpretable) text models often exhibit higher predictive power. Yano et al. (2012) predict whether a bill will survive consideration by U.S. House of Representatives committees, using a logistic regression model. To incorporate the bill texts, they use unigram features indicating the presence of vocabulary terms. Nay (2017) conducted the most comprehensive law-making prediction study to date, which predicted the nearly 70,000 bills introduced in the U.S. Congress from 2001 to 2015.

The only pre-processing applied to the text in Nay (2017) was removal of HTML and carriage returns, and conversion to lower-case. Then inversion of distributed language models was used for classification, as described in Taddy (2015). Distributed language models were separately fit to the sub-corpora of successful and failed bills from past Congresses by applying the word2vec algorithm. Each sentence of a testing bill was scored with each trained language model and Bayes' rule was applied to these scores and prior probabilities for bill enactment to obtain posterior probabilities. The proportions of bills enacted in the same chamber as the predicted bill in all previous Congresses were used as the priors. The probabilities of enactment were then averaged across sentences in a bill to assign an overall probability.

Starting in 2001 with the 107th Congress, I trained models on data from previous Congresses, predicted all bills in the current Congress, and repeated until the 113th Congress served as the test. The model successfully forecast bill enactment: the median of the predicted probabilities where the true outcome was failure (0.01) was much lower than the median of the predicted probabilities where the true outcome was enactment (0.71).

With the language models, in addition to prediction, we can create "synthetic summaries" of hypothetical bills by providing a set of words that capture any topic of interest. Comparing these synthetic summaries across chamber and across Enacted and Failed categories uncovers textual patterns of how bill content is associated with enactment. The title summaries are derived from investigating word similarities within word2vec models estimated on title texts and the body summaries are derived from similarities within word2vec models estimated on the full bill texts. Distributed representations of the words in the bills capture their meaning in a way that allows semantically similar words to be discovered. Although bills may not have been devoted to the topic of interest within any of the four training data sub-corpora, these synthetic summaries can still yield useful results because the queried words have been embedded within the semantically structured vector space along with all vocabulary in the training bills. For instance, I investigated the words that best summarize "climate change emissions", "health insurance poverty", and "technology patent" topics for Enacted and Failed bills in both the House and Senate (Fig. 10). "Impacts," "impact," and "effects" are in House Enacted while "warming," "global," and "temperature" are in House Failed, suggesting that, for the House climate change topic, highlighting potential future impacts is associated with enactment while emphasizing increasing global temperatures is associated with failure. For the health insurance poverty topic, "medicaid" and "reinsurance" are in both House and Senate Failed. The Senate has words related to more specific health topics, e.g. "immunization" for

Failed and "psychiatric" for Enacted. For the patent topic, "software" and "computational" are in Failed for the House and Senate, respectively.

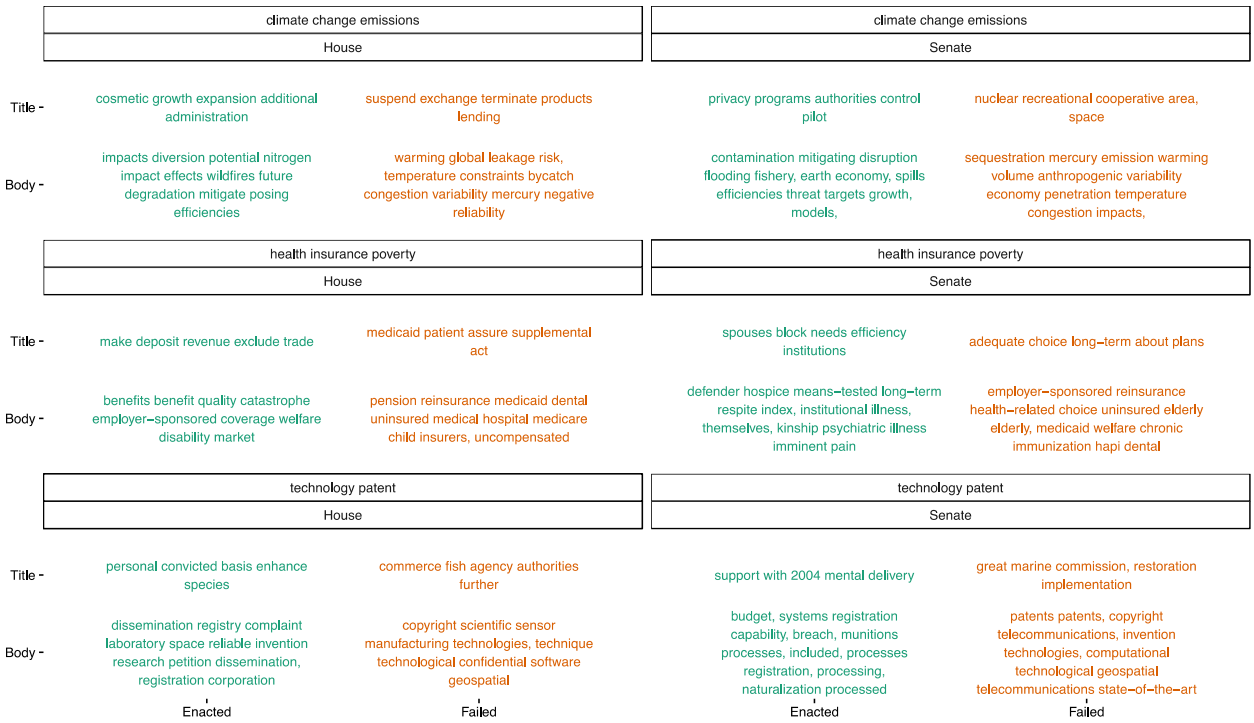


Fig. 10. Synthetic summary bills for three topics across Enacted and Failed and House and Senate categories.

The text model provides sentence-level predictions for an overall bill and thus predicts what sections of a bill may be the most important for increasing or decreasing the probability of enactment. Fig. 11 compares patterns of predicted sentence probabilities as they evolve from the beginning to the end of bills across four categories: enacted and failed and the latest available and first available bill texts. In the latest available (newest) texts of enacted bills, there is much more variation in predicted probabilities within bills.

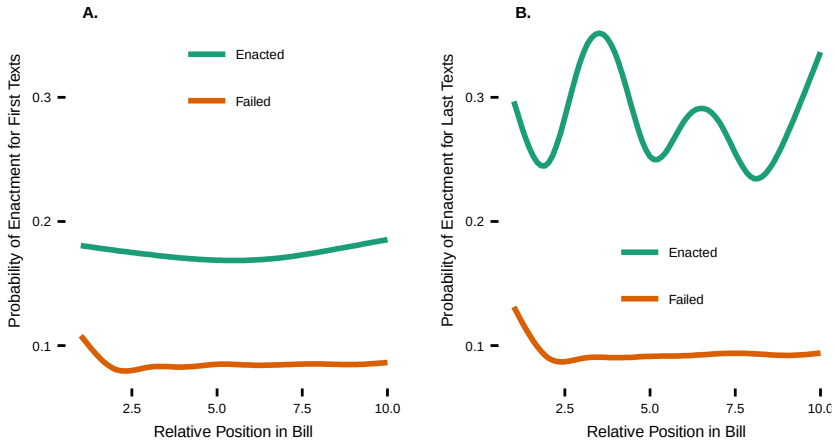


Fig. 11. Sentence probabilities across bills for oldest data (A.) and newest data (B.). For each bill, we convert the variable length vectors of predicted sentence probabilities to *n*-length vectors by sampling *n* evenly-spaced points from each bill. We set *n=10* because almost every bill is at least 10 sentences long. Then we loess-smooth the resulting points across all bills to summarize the difference between enacted and failed and newest and oldest texts.

4.5. Answering Questions

There are two approaches to answering questions automatically. The simpler – and at this time more effective – approach is to learn a function that attempts to recall the most likely answer from a predefined set of answers based on the pairs of answers and questions the model has been trained to recall (Feng et al. 2015). The second approach uses complex models that can actually generate words in new and creative ways rather than relying on predefined answer sets. This uses similar language generation techniques as the tools for generating abstractive summaries. Based on past examples answers and questions, the model learns to encode a question into mathematical space and then decode that representation into a sequence of words that addresses the question. Question

answering techniques could be used to design "chat-bots" that answer legal simple questions by training models on past examples of legal questions and answers.

5. Conclusion

This Paper provided a high-level overview of NLP tools and techniques applied to legal informatics. We provided an introduction to machine learning and its uses in state-of-the-art modeling applications. As we described tools for summarizing textual content and predicting outcomes correlated with textual data, we provided multiple in-depth examples to illustrate the power of machine learned NLP models. There is significant potential for academic studies to use these techniques for summarizing patterns of law across vast amounts of text and for detecting how laws change over time and jurisdiction. Perhaps more significant are the efficiency gains that legal services providers may realize by embedding these computational techniques in work-flows that augment the synthesizing and reasoning skills of attorneys.

References

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., … Collins, M. (2016). Globally Normalized Transition-Based Neural Networks. *arXiv:1603.06042 [cs]*. Retrieved from http://arxiv.org/abs/1603.06042

Airoldi, E. M., & Bischof, J. M. (2015). A regularization scheme on word occurrence rates that improves estimation and interpretation of topical content. *Journal of the American Statistical Association*, *0*(ja), 00–00. http://doi.org/10.1080/01621459.2015.1051182

Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A Neural Probabilistic Language Model. *J. Mach. Learn. Res.*, *3*, 1137–1155.

Blei, D. M. (2012). Probabilistic Topic Models. *Commun. ACM*, *55*(4), 77–84. http://doi.org/10.1145/2133806.2133826

Blei, D. M., & Lafferty, J. D. (2006). Dynamic Topic Models. In *Proceedings of the 23rd International Conference on Machine Learning* (pp. 113–120). New York, NY, USA: ACM. http://doi.org/10.1145/1143844.1143859

Blei, D. M., & Lafferty, J. D. (2007). A Correlated Topic Model of Science. *The Annals of Applied Statistics*, *1*(1), 17–35.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, *3*, 993–1022.

Cardie, C., Farina, C., & Bruce, T. (2006). Using Natural Language Processing to Improve eRulemaking: Project Highlight. In *Proceedings of the 2006 International Conference on Digital Government Research* (pp. 177–178). San Diego, California, USA: Digital Government Society of North America. http://doi.org/10.1145/1146598.1146651

Carenini, G., Cheung, J. C. K., & Pauls, A. (2013). MULTI-DOCUMENT SUMMARIZATION OF EVALUATIVE TEXT. *Computational Intelligence*, *29*(4), 545-576.

Cormack, G. V., & Grossman, M. R. (2014). Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (pp. 153-162). ACM.

Dodds, P. S., Harris, K. D., Kloumann, I. M., Bliss, C. A., & Danforth, C. M. (2011). Temporal Patterns of Happiness and Information in a Global Social Network: Hedonometrics and Twitter. *PLoS ONE*, *6*(12), e26752. http://doi.org/10.1371/journal.pone.0026752

Eisenstein, J., O'Connor, B., Smith, N. A., & Xing, E. P. (2010). A Latent Variable Model for Geographic Lexical Variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing* (pp. 1277–1287). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=1870658.1870782

Feng, M., Xiang, B., Glass, M. R., Wang, L., & Zhou, B. (2015). Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (pp. 813–820). http://doi.org/10.1109/ASRU.2015.7404872

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (pp. 363–370). Stroudsburg, PA, USA: Association for Computational Linguistics. http://doi.org/10.3115/1219840.1219885

Gerrish, S. M., & Blei, D. M. (2011). Predicting legislative roll calls from text. In *In Proc. of ICML*.

Gerrish, S. M., & Blei, D. M. (2012). How They Vote: Issue-Adjusted Models of Legislative Behavior. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 2753–2761). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/4715-how-they-vote-issue-adjusted-models-of-legislative-behavior.pdf

Grossman, M. R. & Cormack, G. V. (2016). Continuous Active Learning for TAR. Retrieved from http://us.practicallaw.com/w-001-8253

Kennedy, A., & Inkpen, D. (2006). SENTIMENT CLASSIFICATION of MOVIE REVIEWS USING CONTEXTUAL VALENCE SHIFTERS. *Computational Intelligence*, *22*(2), 110–125. http://doi.org/10.1111/j.1467-8640.2006.00277.x

Kogan, S., Levin, D., Routledge, B. R., Sagi, J. S., & Smith, N. A. (2009). Predicting Risk from Financial Reports with Regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 272–280). Stroudsburg, PA,

USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=1620754.1620794

Kwon, N., Shulman, S. W., & Hovy, E. (2006). Multidimensional Text Analysis for eRulemaking. In *Proceedings of the 2006 International Conference on Digital Government Research* (pp. 157–166). San Diego, California, USA: Digital Government Society of North America. http://doi.org/10.1145/1146598.1146649

Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., & Jurafsky, D. (2013). Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. *Computational Linguistics*, *39*(4), 885–916. http://doi.org/10.1162/COLI_a_00152

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval* (1 edition). New York: Cambridge University Press.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *ACL (System Demonstrations)* (pp. 55-60).

Mcauliffe, J. D., & Blei, D. M. (2008). Supervised Topic Models. In J. C. Platt, D. Koller, Y. Singer, & S. T. Roweis (Eds.), *Advances in Neural Information Processing Systems 20* (pp. 121–128). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/3328-supervised-topic-models.pdf

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into texts. *Association for Computational Linguistics*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 26* (pp. 3111–3119). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Mohammad, S. M., & Turney, P. D. (2010). Emotions Evoked by Common Words and Phrases: Using Mechanical Turk to Create an Emotion Lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text* (pp. 26–34). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=1860631.1860635

Nay, J. J. (2016). "Gov2Vec: Learning Distributed Representations of Institutions and Their Legal Text." *Proceedings of 2016 Empirical Methods in Natural Language Processing Workshop on NLP and Computational Social Science*, 49–54, Association for Computational Linguistics.

Nay, J. J. (2017). "Predicting and Understanding Law-Making with Word Vectors and an Ensemble Model." *PLoS ONE* 12(5): e0176999.

Ruhl, J.B., Nay, J. J., Gilligan, J.M. (2018). "Topic Modeling the President: Conventional and Computational Methods." *George Washington Law Review.*

Nielsen, F. Arup. (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on "Making Sense of Microposts": Big things come in small packages.* 93-98. Retrieved from http://arxiv.org/abs/1103.2903

Owsley, S., Sood, S., & Hammond, K. J. (2006). Domain Specific Affective Classification of Documents. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs* (pp. 181-183).

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs Up?: Sentiment Classification Using Machine Learning Techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10* (pp. 79–86). Stroudsburg, PA, USA: Association for Computational Linguistics. http://doi.org/10.3115/1118693.1118704

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, *2*(1-2), 1-135.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137. http://doi.org/10.1108/eb046814

Read, J. (2005). Using Emoticons to Reduce Dependency in Machine Learning Techniques for Sentiment Classification. In *Proceedings of the ACL Student Research Workshop* (pp. 43–48). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=1628960.1628969

Roberts, M. E., Stewart, B. M., Tingley, D., Airoldi, E. M., & others. (2013). The structural topic model and applied social science. In *Advances in Neural Information Processing Systems Workshop on Topic Models: Computation, Application, and Evaluation*.

Roitblat, H. L., Kershaw, A., & Oot, P. (2010). Document categorization in legal electronic discovery: computer classification vs. manual review. *Journal of the American Society for Information Science and Technology*, *61*(1), 70-80.

Rosen-Zvi, M., Chemudugunta, C., Griffiths, T., Smyth, P., & Steyvers, M. (2010). Learning Author-topic Models from Text Corpora. *ACM Trans. Inf. Syst.*, *28*(1), 4:1–4:38. http://doi.org/10.1145/1658377.1658381

Russell, S., & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach* (3rd edition). Upper Saddle River: Prentice Hall.

Taddy, M. (2015). Document Classification by Inversion of Distributed Language Representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics* (Vol. Short Papers, pp. 45–49). Beijing, China: Association for Computational Linguistics.

Thomas, M., Pang, B., & Lee, L. (2006). Get out the Vote: Determining Support or Opposition from Congressional Floor-debate Transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* (pp. 327–335). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=1610075.1610122

Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* (pp. 173–180). Stroudsburg, PA, USA: Association for Computational Linguistics. http://doi.org/10.3115/1073445.1073478

Yano, T., Smith, N. A., & Wilkerson, J. D. (2012). Textual Predictors of Bill Survival in Congressional Committees. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 793–802). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from http://dl.acm.org/citation.cfm?id=2382029.2382157