Language Technologies for Understanding Law, Politics, and Public Policy

by

William P. Li

B.A.Sc. Engineering Science, University of Toronto (2009)
S.M. Electrical Engineering and Computer Science,
Massachusetts Institute of Technology (2012)
S.M. Technology and Policy,
Massachusetts Institute of Technology (2012)

Submitted to the
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2016

Author .....
Department of Electrical Engineering and Computer Science
January 15, 2016

Certified by .....
Andrew W. Lo
Professor
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Language Technologies for Understanding Law, Politics, and Public Policy

by

William P. Li

Submitted to the Department of Electrical Engineering and Computer Science
on January 15, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

This thesis focuses on the development of machine learning and natural language processing methods and their application to large, text-based open government datasets. We focus on models that uncover patterns and insights by inferring the origins of legal and political texts, with a particular emphasis on identifying text reuse and text similarity in these document collections. First, we present an authorship attribution model on unsigned U.S. Supreme Court opinions, offering insights into the authorship of important cases and the dynamics of Supreme Court decision-making. Second, we apply software engineering metrics to analyze the complexity of the United States Code of Laws, thereby illustrating the structure and evolution of the U.S. Code over the past century. Third, we trace policy trajectories of legislative bills in the United States Congress, enabling us to visualize the contents of four key bills during the Financial Crisis. These applications on diverse open government datasets reveal that text reuse occurs widely in legal and political texts: similar ideas often repeat in the same corpus, different historical versions of documents are usually quite similar, or legitimate reasons for copying or borrowing text may exist. Motivated by this observation, we present a novel statistical text model, Probabilistic Text Reuse (PTR), for finding repeated passages of text in large document collections. We illustrate the utility of PTR by finding template ideas, less-common voices, and insights into document structure in a large collection of public comments on regulations proposed by the U.S. Federal Communications Commission (FCC) on net neutrality. These techniques aim to help citizens better understand political processes and help governments better understand political speech.

Thesis Supervisor: Andrew W. Lo
Title: Professor

Acknowledgments

I may have clicked my computer's keys in a previously unseen, carefully ordered sequence of characters to write this thesis, but it really takes a village to create a dissertation. I owe my gratitude to many people:

Advisors. My PhD committee shaped and strengthened this thesis: First, Andrew Lo's zeal for big, ambitious ideas drove the interdisciplinary nature of this work — he encouraged me to think beyond the traditional boundaries of computer science and take on fascinating research questions, from unmasking the authors of unsigned Supreme Court opinions to analyzing the United States Code as a large software codebase. Second, Bob Berwick guided me on computational linguistics and natural language processing, and his insights at key points in the research process helped give this thesis rigor. He was always available, often on short notice, to discuss my progress and to provide encouragement. Finally, it was my good fortune to take Ethan Zuckerman's Future of News and Participatory Media class in the MIT Media Lab, both for the opportunity to learn from Ethan and for his kind, compassionate mentorship and support in my last two years at MIT. This work was made possible only through the support of these three remarkable, accomplished human beings.

I also thank my master's co-advisors for their continued support and mentorship. Nick Roy and his Robust Robotics Group introduced me to machine learning and statistical modeling, and I would not be where I am today without Nick's high expectations and and continued guidance. With his patience and gentle approach, Jim Glass taught me about automatic speech recognition, running experiments, and why speech is an excellent substrate for learning about statistical modeling, artificial intelligence, and machine learning; one spectrogram at a time, Jim ignited my interest in the computational study of language and speech. Last, but certainly not least, I thank Seth Teller, whose energy and ethos as a researcher, teacher, and mentor in his life have had a profound personal and professional impact. Nick, Jim, and Seth: Thank you.

Collaborators. I am fortunate to have had an unusually large number of research

co-authors during my time at MIT: Pablo Azar, David Larochelle, Phil Hill, Jay Cox, Finale Doshi-Velez, Emad William, Descartes Holland, Jason Chuang, Sands Fish, Rebecca Weiss, Erika Lee, Minh Phan, Alex Sekula, Sophie Chou, Ramesh Sridharan, Yoni Battat, Jun-geun Park, Dorothy Curtis, Sachithra Hemachandra, Bryan Reimer, Javier Velez, Cynthia Walsh, Don Fredette, Alex Burnham, Bob Lamoureux, Marva Serotkin, Simone Fontanesi, Alessandro Frigerio, and Nick Wang. I also learned from a great set of internship supervisors: John Hershey, Daniel Nikovski, Belle Tseng, Gunnar Evermann, and Larry Gillick. In particular, I wish to highlight Pablo, David, and Phil's roles as the nucleus of the two law review papers that make up the first part of this thesis; the thousands of hours I spent with Sachi in 32-337 and his help with getting me started as a graduate student; Don's instrumental role in my work with the Boston Home, both in research and in many MIT assistive technology efforts; and Finale's guidance and mentorship in both my first project in 2009 (on improving resident safety using Wi-Fi localization at the Boston Home) and my last one in 2016 (on Probabilistic Text Reuse). I have learned so many things from every one of these collaborators, including how to solve problems and do effective research, good coding and software engineering practices, core concepts in statistics, machine learning, and natural language processing, and much more from their deep expertise in fields as diverse as assistive technology, health care, and law.

The opportunity to co-teach the Fall 2014 offering of 6.811: Principles and Practice of Assistive Technology (PPAT) was both the most challenging and most rewarding part of my experience at MIT. I am deeply thankful that Rob Miller entrusted me to co-lead the course and offered his mentorship as an accomplished computer science educator, as well as for the support of EECS, the d'Arbeloff Fund for Excellence in Education, and MIT OpenCourseWare for sustaining Seth Teller's vision for assistive technology education at MIT. The course would not have been possible without our extraordinary team, including Grace Teo, Michelle Chen, Ishwarya Ananthabhotla, Abigail Klein, and Jeff Dusek, as well as the new team in Fall 2015: Grace, Jeff, Ishwarya, Wenxin Feng, Len Evenchik, Matthew Schneps, and both Julie Greenberg and John Leonard for ensuring the continued success of the course in 2016 and beyond.

Finally, I thank Rachel Zimmerman for her strength and support for the class and for assistive technology at MIT.

Supporters. Dozens of MIT staff and faculty have had major positive roles in my experience at MIT, both within and outside of the world of research. They include Bryt Bradley, Sophia Hasenfus, Marcia Davidson, Jayna Cummings, Patsy Thompson, Allie McDonough, Janet Fischer, Elizabeth Bruce, Pattie Maes, Sam Madden, Alison Hynd, Mary Ziegler, Kathleen Cahill, Sydney Miller, Ed Ballo, Roger and Dottie Mark, Annette Kim and Roland Tang, Andreas Schulz and Berit Johannes, Julie and Neel Shah, Steve Ward, Dava Newman, and Leslie Kolodziejski. Their helpfulness, generosity, and insight contributed immensely positively to my experience in graduate school, my well-being, and my ability to get to this point today.

Communities. Graduate school has been about much more than research; the communities and teams I have been a part of have shaped my life at MIT for the better. Sidney Pacific, TPP, TBH, RRG, RVSN, SLS, CSAIL/EECS, LFE, CoSI, ATHack, PSC, Law Is Code, and the Berkman Center have all been a sources of friendships and shared experiences.

Friends. I feel fortunate to count the people listed above not only as professional collaborators and mentors, but also as friends. In addition, though, there have been many people who I need to acknowledge for their support throughout grad school. They include TPPers like Pam DeAmicis, Rachna Pande and Akhil Basha (and Theo), Rubén Garcia, James Merrick, Francisco Alonso, Judy Wang, Tommy Leung, and Nathan Perkins; SPers like David Kwabi, Boris Braverman, George Chen, Rachael Harding, Mirna Slim, Armen Mkrtchyan, Kelli Xu, Brian Spatocco, Stephen Morgan, Lily Xu, William Li (not a typo), George Lan, Stephanie Nam, Bernhard Zimmermann, Holly Johnsen, George Tucker, Ramesh Sridharan, Audrey Fan, Tarun Jain, Vishnu Desaraju, Diana Chien, Matt D'Asaro, Steven Chang, Danica Chang, Frank Wang, and Michael Peng; Stephen Shum, Ekapol Chuangsuwanich, David Harwath, Ann Lee, Steve Levine, Sudeep Pillai, Ross Finman, David Hayden, Javier Velez, Josh Joseph, Charlie Richter, Elena Glassman, Juho Kim, Amy Zhang, Adam Marcus, Eugene Wu, and many others in CSAIL; and friends from across MIT and other places,

including William Palin, David Colarusso, Karn Saroya, Anand Dhillon, Alfred Chan, Kelvin So, and Sahitya Gupta.

Among all of these friends and colleagues, Carrie Cai deserves her own paragraph — this thesis would not have been possible without her. In addition to being a source of regular encouragement, it is worth noting that Carrie has contributed substantially to many aspects of my life, including long discussions about research, professional development, career opportunities, and even a shared interest in topics ranging from social sciences and the humanities to the Asian-American experience. I could not have asked for a better friend, ally, and partner through the roller-coaster ups and downs of graduate school.

Family. Last, but not least, I thank my family — my sister Joyce, my Mom, and my Dad — for helping me get here. By example, my parents taught me the value of hard work and integrity, and they gave me the support, independence, and confidence to succeed. I dedicate this thesis to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

How do we, as a society, collectively make decisions? Who writes our laws, regulations, and judicial opinions? What are the impacts of the policy choices that governments make? Given its profound impacts on our lives, governments and political processes are fascinating and essential to study; moreover, an unprecedented amount of data on legal and political processes, is easily available for download and analysis. The ambition of this thesis is to develop and apply computational techniques — algorithms, applications, and systems — that make sense of the large collections of public data on government and political processes.

1.1 Motivation

This thesis focuses on the development and application of "civic" technology: The collection of projects aim to illustrate how language technologies, particularly the analysis and understanding of large public text datasets, can promote the common good. To that end, we focus on two overarching goals:

1. **To help citizens better understand government processes:** Governments are large, multi-dimensional organizations charged with managing and addressing complex, society-scale problems. In modern liberal democracies, we delegate policymaking to elected professional representatives and the execution

of laws to large groups of civil servants; however, in order to make informed choices and to keep institutions accountable, it seems reasonable that citizens should understand government processes. Our work on Supreme Court authorship attribution (Chapter 2), analyzing the complexity of the law using software engineering metrics (Chapter 3), and identifying the trajectories of policy ideas in Congress (Chapter 4) show how the analysis of large government text corpora can reveal useful insights into these legal and political processes.

2. **To help government better understand public speech:** In democratic societies, governments should understand and serve the preferences of the people they represent. In small societies, such as ancient Athens or today's small towns and communities, hearing all of these voices is feasible, perhaps in the form of physical gatherings; in larger contexts, a central challenge is capturing and understanding public opinion from millions or billions of citizens. Today, governments have elaborate apparatuses, from elections to polls and petitions, to measure the will of the people; notably, however, these measurement methods largely involve reducing citizen voices to counts of pre-ordained ideas and viewpoints. Understanding the rich diversity of ideas and opinions by members of society, perhaps through text or other forms of language, is both a challenge and an opportunity that this thesis seeks to tackle. Chapter 5 introduces a novel computational model, called Probabilistic Text Reuse (PTR), and applies it to a real-world example of a large collection of public speech: nearly a million public comments on the U.S. Federal Communication Commission's proposed 2014 regulations on Net Neutrality.

1.2 Background

The work in this thesis is situated at the intersection of 1) the unprecedented amount of government data available for analysis, 2) the rise of the field of data science and its accompanying techniques and tools, and 3) the unique position of computer scientists, in collaboration with stakeholders and researchers in other disciplines, to

tackle these problems and catalyze further activity in this domain. We discuss each of these developments in turn.

1.2.1 Opportunity: Big Data and E-Government

Government has become bigger and more complex, with deeper involvement in all many aspects of society. As just one example, Figure 1-1 shows that, by 2011, the size of the United States Code of Laws had increased to nearly 16 times its length since 1925, when it was first introduced. This enormous growth reflects the complexity of the federal government's role, from healthcare and environmental protection to financial regulation and space exploration. It is simply impractical to read all of the laws, regulations, or other documents generated by government.

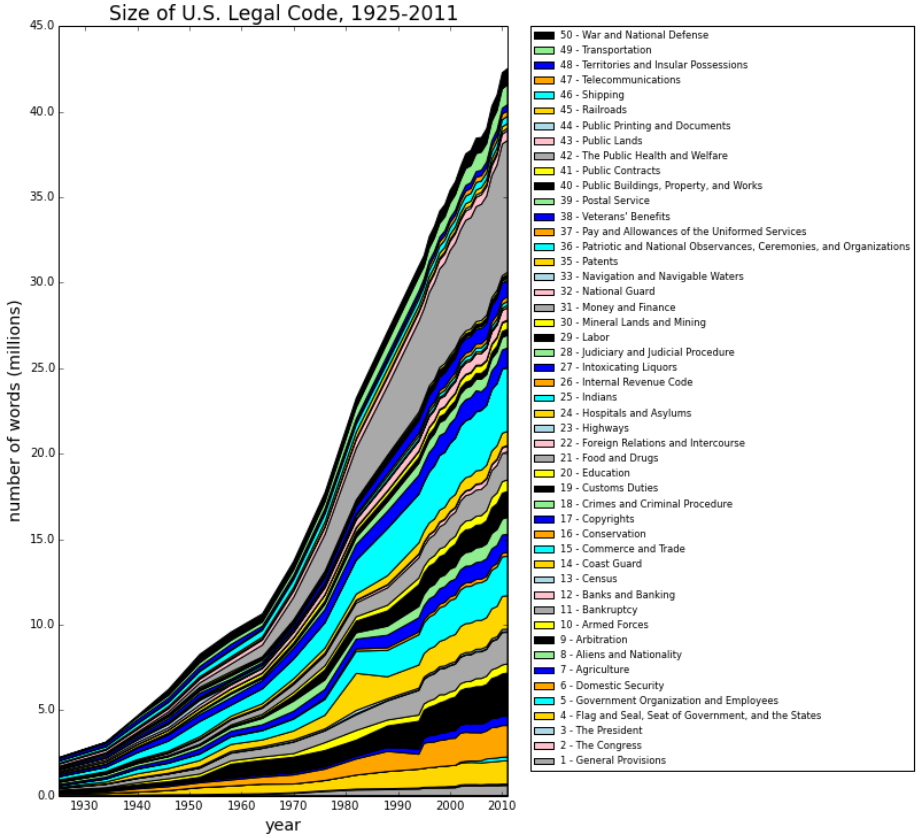


Figure 1-1: Size of United States Code, 1925-2011

In addition to the increase in the sheer size of government, an unprecedented amount of this data is now publicly available. Many of these efforts are initia-

tives of the federal government itself; portals such as the Government Printing Office (GPO) and `data.gov` have made other datasets available [32]. In addition, efforts by civil society groups and journalists have liberated more data and made it accessible for novel applications, from the Sunlight Foundation's work to track legislation through Congress [3] to documents released by Freedom of Information Act (FOIA) requests. In another era, such data might be locked in file cabinets, libraries, or physical archives; today, collecting this data from different digital sources can require much less effort.

Finally, the issues faced by government and society are growing in complexity. To take banking and finance laws as one example, the Financial Crisis of 2007-2009 produced major legislative responses, from the Troubled Asset Relief Program (TARP) in 2008 to the Dodd-Frank Wall Street Reform and Consumer Protection Act of 2010. The intricacies of the financial sector, coupled with the nuances of the legislative response, make it difficult for even the most informed citizen to understand the implications of the government's actions. Similar arguments can be made about the complexity of the regulations promulgated by federal agencies or the issues tackled by the judicial system. For these reasons, new, more scalable approaches to understanding government are needed.

1.2.2 Technology: Data Science for Large Document Collections

To address the challenges outlined in the previous section, this research uses tools and techniques from machine learning and statistical natural language processing. Advances in supervised learning, structured prediction [e.g., 51], and unsupervised algorithms such as probabilistic topic modeling [e.g., 7] or computing vector representations of words [34].

In this thesis, the objective in each of the presented projects is to derive useful insights from legal and political documents. The recent rise of the discipline of **data science** provides a model for our work: In addition to developing machine learning

models, the chapters of this thesis all involve real-world datasets and a discussion of the implications of the findings for diverse stakeholders and audiences. The full process of each chapter is as follows:

1. Data: Finding and preparing the dataset for the task is the first, and often most important, step in a data science research project. While this thesis is fundamentally motivated by the unprecedented public availability of government data, choosing the right dataset for the task and structuring the data into a usable form are important decisions with downstream analysis implications.

2. Task Definition: The next task is to refine the goal of "discovering insights" into an appropriately scoped, achievable objective. In some cases, a quantitative loss function can be minimized; in other situations, the goal may be sufficient to find interesting patterns in the dataset.

3. Features and Models: Often the focus of academic machine learning, computing features and developing a model is explicitly just one component of the approach taken in this thesis to solve problems.[1]

4. Evaluation: Choosing the right model requires some criteria comparing competing models. Determining the right metric for the specific problem being solved is central to the goal of understanding government using machine learning.

5. Communication and Visualization: Presenting the results in a meaningful, accessible manner, especially to non-experts, is not an afterthought. Being able to communicate the results effectively could make the difference between wider adoption of the work of this research and having it simply languish as a theoretical research result.

[1]The work "Machine Learning That Matters" by Wagstaff [53] is well worth reading and provides a more in-depth discussion of this point.

Fit: A Need for Data Science on Open Government Datasets

While the new availability of large public datasets offers a window into modern government, this potential cannot be unlocked without the right kinds of analysis. In particular, large, unstructured text datasets often require different algorithms, systems, and skills that those that are currently available. Similarly, these skillsets are often lacking in resource-constrained media and watchdog organizations that seek to promote government understanding and transparency [12].

In general, media and civil society organizations have begun to build interesting applications on top of government data to make it more accessible [59]. In 2009, for example, researchers developed RECAP, a Firefox extension that crowd-sources the free availability of court documents from the pay-per-use PACER online law database [15]. Meanwhile, the Sunlight Foundation built application programming interfaces (APIs) to make congressional activities easier to access. Few applications, however, involve sophisticated data science techniques and analysis of these datasets. Building on these early efforts, this dissertation aims to add value through the analysis and visualization of these text-based datasets, and to lower the barriers that prevent other researchers, activists, and government itself from adopting these technologies.

1.3 Related Work

This section begins with a description of advances in making government data available. It then provides a summary of the methodological advances and noteworthy applications related to the research goals described in this thesis.

1.3.1 Open Government Data

Through its many activities, the U.S. government leaves a large "paper trail" of text data as it conducts its activities. Audio or video recordings of legislative, judicial, or regulatory proceedings are often kept, and the government undertakes the expense of producing complete transcripts. Today, much of this data is freely available for

download online.

Beyond this surface description, however, there are important caveats. One limitation is that not all interesting datasets are publicly available: some activities are not recorded, some are classified, some are incomplete, and some are not made public by default. Even if the desired dataset existed, however, the data is often highly heterogeneous and difficult to process. The federal government alone has hundreds of agencies, each with its own reporting formats and standards, and current and historical data may be available only in print, as scanned images, or as difficult-to-process PDFs or other digital forms. These barriers present challenges and barriers to applying machine learning algorithms to these datasets.

To address these issues, a number of parties have proposed paradigms or initiatives that address these problems for different datasets. To begin, Robinson et al. [44] argue that, as an information technology policy, government should expose the underlying data and allow third parties to provide services and insights. It would allow both government agencies and private parties to build innovative presentations and analysis of the data. Practically, being able to download and access data in bulk would solve the problem of having to scrape government websites, a task that can range from highly tedious to technically infeasible. Among other benefits, the authors argue that, by exposing the underlying data, outside parties could develop more advanced features, including "mashups with other data sources," "visualization", and "automated content and topic analysis" [44]. This thesis seeks to accelerate the development and adoption of technologies for government datasets.

In the United States, some noteworthy open data projects from governments and civil society organizations include:

- The Cornell Legal Information Institute (LII) was founded in 1992 and is a pioneer of making legal information available in a structured format [9]. Currently, documents ranging from the Constitution to Supreme Court decisions are available in machine-readable formats [8].

- 2004 saw the launch of GovTrack, which screen scrapes official government

websites and makes the data available in a structured database. Currently, information about members of Congress, bills and resolutions, voting records, and committee activity are available from http://www.govtrack.us in a structured form that can be accessed via an API or as a bulk download. As of June 2014, 34 other open government websites use GovTrack as their data platform [52].

- Open government advocates created the "8 Principles of Open Government Data" at a workshop in December 2007 [1]. The eight principles are that data should be 1) complete, 2) primary, 3) timely, 4) accessible, 5) machine processable, 6) non-discriminatory, 7) non-proprietary, and 8) license-free. The availability of data that complies with these principles underpins the research that this thesis will contain.

- The administration of President Barack Obama has taken an interest in open government, with mixed success. In January 2009, the Administration issued an executive memorandum on "Transparency and Open Government" that called for the federal government to be more transparent, participatory, and collaborative [40]. This memo was followed by the Open Government Directive in December 2009, which called for government agencies to form open data plans and to post datasets on the newly established Data.gov website Data.gov [42], which has had mixed success; many open government advocates applauded the idea, but noted that many datasets were incomplete, out of date, or only focused on non-controversial issues [32]. An Executive Order in 2013 that called for government information to be open and machine readable led to changes in Data.gov, including more powerful search tools and APIs to access data [41].

- The OpenGov Foundation and the StateDecoded Project have focused on making proposed laws and existing legal codes machine readable and accessible in more user-friendly, interactive formats. For example, the first StateDecoded project, VirginaDecoded, includes hyperlinks to definitions and makes the laws of Virginia searchable [2]. The Madison Project, meanwhile, focuses on allowing citizens to participate in the lawmaking process by inviting the public to

comment on policy documents and even crowd-source new laws. In both cases, the availability of the text of the law in a structured, machine-readable format enables these projects, along with other, potentially more advanced analyses of the text.

- The Sunlight Foundation is dedicated to government transparency, making it a key advocate for making government data openly available. A portion of their work includes a number of tools focused on tracking influence, the activities of Congress [50].

- The Congressional Bills Project is an annotated database of bills from 1947 to the present [4]. Bill sponsor information, relevant event dates, and the original bill text segmented by section are all available. Recently, data from the project was used in the 2014 Unshared Task in PoliInformatics, which focused on computational approaches to understanding the 2007-2009 Financial Crisis [11].

NLP Applications on Legal and Political Texts

By analyzing large government datasets, an opportunity exists to advance our understanding of legal and political systems. This section provides a sketch of the goals of broad areas of natural language processing and noteworthy applications in the area of government datasets.

Supervised Learning: Classification and Regression

Given a set of documents that are labeled with a known target variable, supervised learning seeks to make label predictions on unseen test documents. In classification, the possible targets are typically two or more categories; in regression, a continuous value is the target. A myriad of classification algorithms exists, including logistic regression, decision trees, support vector machines, and boosting; each of these algorithms has counterparts for regression. Some noteworthy applications on legal and political texts include:

Authorship Attribution: Mosteller and Wallace applied statistical techniques to infer the authorship of 12 disputed Federalist Papers [37]. Their 1963 paper, which found that James Madison was the most likely author of the papers, was one of the earliest examples of Bayesian inference. Other recent work has focused on the Supreme Court, including efforts to understand the influence of clerks in the writing of judicial opinions [45] and to unmask the authors of unsigned or controversial opinions [27].

Bill Success: Yano, Smith, and Wilkerson demonstrated the potential of text features to predict bill survival in Congressional committees [57].

Political Positions and Affiliations: Laver et al. pioneered efforts to bring statistical techniques to political texts, showing that a bag-of-words approach could successfully predict the movement of the British Labour Party from the left to the center of the political spectrum between 1992 and 1997 [25]. Past work has also included classifying party affiliations and ideologies from political speech [58, 48, 18].

Structure Recovery: A key challenge in many real-world government datasets is that they are not fully annotated, making them less machine-readable and amenable to large-scale analysis. An interesting approach, therefore, is to use automated approaches to try to imbue documents with structure. For example, automatically identifying citations in public comment submissions might be solved by manually labeling a few hundred examples and letting machine learning algorithms learn the appropriate patterns [e.g., 5].

Pattern Discovery and Unsupervised Learning

Insights can also emerge from finding patterns or structure in large collections of legal and political documents. For example, using text similarity approaches, Wilkerson et al. showed the trajectory of bill sections in the 2009 Patient Protection and Affordable Care Act (PPACA) and other bills [56]. Similar techniques have also been applied to estimate policy trajectories and consideration times of bills related to the 2007-2009 Financial Crisis [28].

1.4 Research Hypothesis

The central hypothesis unifying this dissertation is:

Machine learning techniques that focus on inferring the origins of text-based legal and political datasets can reveal novel insights into government processes and public speech.

Specifically, approaches that find hidden sources or structure from these datasets are well-suited to uncovering patterns in open government datasets, including Supreme Court opinions, the United States Code, and Congressional bills. The final chapter of this thesis focuses on one particular methodology, text reuse, and develop a novel algorithmic approach and application to public comments on government regulations.

1.5 Roadmap

This thesis begins with three projects on diverse text datasets. First, in Chapter 2, we present an authorship attribution model on unsigned U.S. Supreme Court opinions, offering insights about the authorship of important cases and the dynamics of Supreme Court decision-making. This model makes predictions on a case of particular interest (the authors of the majority and dissenting opinions of the controversial 2012 *NFIB v. Sebelius* ("Obamacare") decision, as well as reveals trends about the authorship of unsigned "per curiam" opinions in the current Supreme Court, as led by Chief Justice John Roberts.

Second, in Chapter 3, we apply concepts and metrics from software engineering to analyze the current and historical complexity of all laws in the United States Code, revealing the structure and evolution of the U.S. Code over the past 100 years. The rise of specific terms and concepts in the U.S. Code, changes in the content of different areas of federal law, and the differences in the structure of various kinds of legislation emerge from this analysis.

Chapter 4 traces policy trajectories of bills in Congress, making it possible to visualize the contents of four key bills during the Financial Crisis and when ideas

in these bills were first introduced. By applying text reuse techniques, it becomes possible to identify some key contents of these bills, when policy ideas first emerged, and the "consideration time" of these ideas in Congress before they were passed into legislation.

Motivated by the commonalities of these three datasets and research questions, the final part of this thesis focuses on techniques and applications for finding text reuse in government datasets. Text reuse occurs in legal and political documents because documents present similar ideas, different versions of documents are often quite similar, and because legitimate reasons for copying text exists. Chapter 5 describes how text reuse occurs legitimately in legal and political contexts — unlike plagiarism, text reuse is often a reasonable approach to writing in legal and political contexts, or a perfectly understandable artifact of government activities. This chapter presents Probabilistic Text Reuse (PTR), a generative model for how reused passages of text occur in a large corpus of documents, and tractable methods for inferring the parameters of the model from real data. Then, this chapter illustrates the utility of PTR and other text reuse methods through the analysis of common text reuse that occurs in public comments on the Federal Communication Commission's proposed regulations on net neutrality.

This thesis concludes in Chapter 6 by speculating on areas of potential future work on data science with open government datasets.

Chapter 2

Authorship Attribution of Unsigned Supreme Court Opinions

Preamble

This chapter is largely adapted from the following paper:

Li, W., Azar, P., Larochelle, D., Hill, P., Cox, J., Berwick, R. C., Lo, A. W. Using Algorithmic Attribution Techniques to Determine Authorship in Unsigned Judicial Opinions. Stanford Technology Law Review, 16, 503-533, June 2013 [27].

2.1 Introduction

U.S. courts publish a shocking number of opinions without divulging the author. Unsigned per curiam opinions, as traditionally and popularly conceived, are a means of quickly deciding uncontroversial cases in which all judges or justices are in agreement. Today, however, unsigned per curiam opinions often dispose of highly controversial issues, frequently over significant disagreement within a court. Obscuring authorship removes the sense of accountability for each decision's outcome and the reasoning that led to it. Anonymity also makes it more difficult for scholars, historians, practitioners, political commentators, and — where applicable — the electorate, to glean valuable information about legal decision-makers and the way they make their decisions. The

value of determining authorship for unsigned opinions has long been recognized but, until now, the methods of doing so have been cumbersome, imprecise, and altogether unsatisfactory. Currently, to obtain information on how decisions were made and authored, the public relies on anecdotal evidence from clerks, legal observers, and occasional comments by judges and justices themselves.

Given the importance of unsigned opinions and the large corpus of signed judicial writings, we demonstrate that novel computational tools can add quantitative, non-partisan insight into judicial opinion authorship. Our work uses statistical data mining and machine learning algorithms to predict authorship of judicial opinions that are unsigned or whose attribution is disputed. Using a dataset of opinions with known authorship, we identify key words and phrases that can, to a high degree of accuracy, predict authorship using only the text from a judicial opinion (with obvious identifying markers removed). After training "writing style models" for the different justices under consideration, we can predict the author of an unsigned opinion by analyzing only that unsigned opinion's text. Our method provides insight into which authors were most influential in writing the published opinion, thereby giving interested parties access to an important class of cases heretofore inaccessible.

Part 2.2 summarizes the historical context of unsigned per curiam opinions, criticisms of the practice, and compares our attribution solution to other approaches. To illustrate our method, the remaining parts of this paper describe the process of determining authorship in a recent, high profile Supreme Court case in which the author of the dissenting opinion was subject to much popular speculation. Part II describes this illustrative test case. Part III describes the experimental setup and Part IV explains the results. Part V provides the results of applying our process to every unsigned per curiam opinion of the Roberts Court.

2.2 Unsigned Opinions

Over the last 150 years, there has been an astonishing number of court decisions issued without attribution. Unsigned opinions have been the subject of great controversy

since their inception and present many problems today. Although unsigned opinions appear in all appellate courts — federal or state — the example set by the Supreme Court is particularly illustrative. Part 2.2.1 provides historical context for unsigned per curiam opinions. Part 2.2.2 summarizes some of the problems that have been associated with judicial anonymity. Part 2.2.3 evaluates some of the current methods used to determine authorship. Part 2.2.4 briefly describes how our method provides a better means of determining authorship.

2.2.1 Historical Context of Unsigned Opinions

The Supreme Court's attribution practices have a long and colorful history.[1] The Court has delivered opinions in one of four ways.[2] First, in the early days, the Court would issue decisions "seriatim," whereby the Justices wrote separate opinions that were published in order of seniority, and were sometimes followed by a summary order "By the Court" with the overall disposition.[3] Second, for uncontroversial and unanimous decisions, the Court would deliver an opinion under the heading "By the Court."[4] Third, the Court would deliver a single opinion under the name of the Chief Justice, while indicating that he was speaking "for the Court."[5] Finally, the Court would issue a majority opinion with justices writing separately as they desired.[6]

The fourth attribution option — a majority opinion accompanied by separate dissents and concurrences — has become the familiar means of delivering opinions.[7] However, during the Marshall era, the Chief Justice chose the third option in an effort to enhance the Court's image of solidarity and authority.[8] Even when the justices disagreed, Marshall insisted that the Court issue only one opinion in his name, even

[1] For a more robust history, see generally John P. Kelsh, The Opinion Practices of the United States Supreme Court 1790-1945, 77 WASH. U. L.Q. 137 (1999); James Markham, Note, Against Individually Signed Judicial Opinions, 56 DUKE L.J. 923, 28 (2006).
[2] Markham, *supra* note 1, at 928.
[3] *Id.*
[4] *Id.*
[5] *Id.*
[6] *Id.*
[7] *Id.* at 29
[8] *Id.*

when he disagreed in the judgment.[9] This practice gradually gave way and by 1832, all members of the Court had written separately at least once. By the time Marshall died in 1835, the practice of writing separate opinions was solidified.[10]

Throughout these shifts in the Court's attributional philosophy, the "per curiam" decision — in which an opinion states the ostensible opinion of "the Court" rather than any particular justice(s) — has remained a viable option. However, such decisions have not been confined to uncontroversial or unanimous topics as they were in the early days of the Court. The classic occasion for a per curiam decision is when the law is so clear that the justices are unanimous and the issue does not merit the time necessary to craft a detailed opinion.[11]

However, with great frequency today's per curiam opinions are neither unanimous nor uncontroversial. A 1992 study found that only 44% of per curiam opinions were unanimous.[12] For the other 56% of per curiam opinions, there are two ineluctable conclusions: (1) despite the label, an ostensibly "per curiam" opinion cannot speak "for the Court" as a whole, and (2) there is at least some controversy to the disposition. Some of the most important and controversial cases in our nation's history came through badly divided per curiam opinions.[13] Such decisions include invalidating the death penalty in Furman v. Georgia (five concurrences, four dissents),[14] dealing with campaign finance reform in Buckley v. Valeo (involving a 137-page per curiam with five opinions concurring in part and dissenting in part),[15] resolving the Pentagon Papers case (six concurrences, three dissents),[16] and ending the presidential election of 2000 in Bush v. Gore (one concurrence, four dissents),[17] to name a few.

---

[9] *Id.*

[10] *Id.*

[11] See, *e.g.*, *id.* at 934; Ira P. Robbins, *Hiding Behind the Cloak of Invisibility: The Supreme Court and Per Curiam Opinions*, 86 TUL. L. REV. 1197, 1200-02 (2012); Laura Krugman Ray, *The Road to* Bush v. Gore*: The History of The Supreme Court's Use of the Per Curiam Opinion*, 79 Neb. L. Rev. 517, 521- 24 (2000); Stephen L. Wasby et al., *The Per Curiam Opinion: Its Nature and Functions*, 76 JUDICATURE 29, 30 (1992).

[12] Wasby et al., *supra* note 11, at 35.

[13] See Michael C. Gizzi & Stephen L. Wasby, *Per Curiams Revisited: Assessing the Unsigned Opinion*, 96 JUDICATURE 110, 113 (2012).

[14] 408 U.S. 238 (1972).

[15] 424 U.S. 1 (1976).

[16] 403 U.S. 713 (1971).

[17] 531 U.S. 98 (2000).

With these qualitative concerns in mind, the number of per curiam opinions is alarming. The Warren Court used per curiam opinions 28.7% of the time, the Berger Court 17.7%, the Rehnquist Court 10.3%, and the Roberts Court 13.3%.[18] In 2011, the federal courts of appeal issued per curiam opinions 7.6% of the time, with significant variation across circuits.[19] Whereas the D.C. Circuit relied on per curiam opinions only 0.3% of the time, the Fifth Circuit used them 15.9% of the time.[20] The problem is direr in some state courts, where per curiam opinions constitute *more than half* of an elected court's decisions.[21]

2.2.2 Problems with Unsigned Opinions

The previous section highlighted the quantity and quality of cases using unsigned opinions, but what is the harm? The most compelling complaints focus on the theme of accountability: poor quality of opinions, evasion of difficult issues, lack of transparency to the public, and the like.[22]

Critics voicing the accountability concern are numerous and frequently high profile. In response to Chief Justice Marshall's edict that the Supreme Court issue a single opinion in his name, Thomas Jefferson wrote that "secret, unanimous opinions" written on behalf of the Court would undermine judicial accountability.[23] When "nobody knows what opinion any individual member gave in any case, nor even that he who delivers the opinion concurred in it himself, [a justice's reputation] is shielded completely."[24] Jefferson disapproved of opinions reached by justices "huddled up in a conclave, perhaps by a majority of one, delivered as if unanimous, and with the silent acquiescence of lazy or timid associates, by a crafty chief judge, who sophisticates

[18] Gizzi & Wasby, supra note 13, at 111.

[19] *Id.* at 114.

[20] *Id.* at 115.

[21] *In the Shadows: A Look into the Texas Supreme Court's Overuse of Anonymous Opinions*, at 1, TEXAS WATCH (May 2008), available at `http://www.texaswatch.org/wordpress/wp-content/uploads/2009/12/PerCuriamReportFinal.pdf`.

[22] See *generally* Robbins, supra note 11. Other critiques include stunting the development of the law by reducing the ability to analyze a judge or justice's jurisprudence and put to use any lessons derived from such analysis. See *id.* 1224-41.

[23] Markham, *supra* note 2, at 930 (quoting Letter from President Thomas Jefferson to Justice William Johnson (Oct. 27, 1820).

[24] *Id.*

the law to his own mind, by the turn of his own reasoning."[25] Per curiam opinions, according to Jefferson, are "certainly convenient for the lazy, the modest, and the incompetent."[26]

Jefferson's disapproval and call for accountability has echoed since. President Madison called for a return to seriatim opinions "so that Republican judges could record their position on the issues."[27] When she was a circuit judge, Justice Ginsburg wrote, "Public accountability through the disclosure of votes and opinion authors puts the judge's conscience on the line."[28] She further noted, "Judges generally do not labor over unpublished judgments and memoranda, or even per curiam opinions, with the same intensity they devote to signed opinions."[29] Approvingly quoting another commentator, Justice Ginsburg wrote, "[W]hen anonymity of pronouncement is combined with security in office, it is all too easy, for the politically insulated officials to lapse into arrogant ipse dixits."[30] Judge Richard Posner agreed, asserting that signed opinions elicit the greatest effort from judges and make "the threat of searing professional criticism an effective check on irresponsible judicial actions."[31] Discussing the Court's decision in *Bush v. Gore*, one commentator noted that per curiams are convenient tools in controversial cases because "[w]ith no Justice signing the opinion, there [is] no individual to be blamed for evading the tough questions."[32]

These accountability criticisms have even more force when the judges and justices are elected officials serving terms of office rather than appointed judges and justices serving for life or good behavior. Thirty-nine states (78%) require judges to run for election or win periodic retention votes.[33] The electorates in these states need recorded votes and opinions to evaluate their respective judges and justices, but

[25] *Id.* (quoting Letter from Thomas Jefferson to Thomas Ritchie (Dec. 25, 1820)).
[26] John P. Kelsh, *The Opinion Practices of the United States Supreme Court 1790-1945*, 77 Wash. U. L.Q. 137, 145-46 (1999) (citing Letter from Thomas Jefferson to William Johnson).
[27] *Id.*
[28] Ruth Bader Ginsburg, Remarks on Writing Separately, 65 WASH. L. REV. 133, 140 (1990).
[29] *Id.* at 139.
[30] *Id.*
[31] Richard A. Posner, *The Federal Courts: Challenge and Reform* 349 (1999).
[32] Laura Krugman Ray, *The Road to* Bush v. Gore*: The History of the Supreme Court's Use of the Per Curiam Opinion*, 79 NEB. L. REV. 517, 521-22 (2000).
[33] Robbins, *supra* note 11, at 1221.

unsigned opinions reduce access to this vital information.

For example, Texas is one state in which judges are elected by and accountable to voters. During the 2006-2007 term, an astounding 57% of the opinions issued by the Supreme Court of Texas were unsigned per curiams.[34] Over a ten-year period, per curiam opinions constituted 40% of the opinions issued by the Supreme Court of Texas.[35] One commentator puts the problem nicely:

> When a judge signs his name to an opinion he has written, he accepts responsibility for the decision and the logic used in reaching it. Whether the opinion is a stellar example of judicial wisdom or a blatant abuse of judicial authority, the author is accountable because his identity is known. Any judge who disagrees with an authored opinion must write or join a dissent, and thus that judge's position is known as well, and he is equally accountable.
>
> When a court releases a per curiam opinion, however, no judge accepts responsibility for the opinion, and no judge can be held accountable for it. The public does not know if all judges agreed with the holding. Judges who can hide behind this anonymity may not have an incentive to reach the legally correct conclusion or to justify the conclusion they do reach.[36]

This same commentator goes on to list controversial per curiam opinions in the Texas Supreme Court and analyzes campaign contributions from parties who appeared before the court in such cases.[37] This example highlights one of the most extreme consequences of judicial unaccountability.

2.2.3 Solving Attributional Questions the Old-Fashioned Way

Scholars and historians have long been skeptical of unsigned opinions and have sounded numerous calls for research identifying the authors of unsigned opinions. Until now,

[34] *In the Shadows*, *supra* note 20.
[35] *Id.*
[36] *Id.*
[37] See *generally id.*

the methodologies recommended and employed have been decidedly "old school."

A 2012 article charting the use of per curiam decisions of the Supreme Court suggested some methods for determining authorship.[38] First, the article recommends narrowing the possibilities by ruling out authors of separately signed opinions.[39] However, this method will frequently yield incomplete results. At best, it narrows the possibilities to the handful of justices who did not write a separate opinion. However, this approach does not rule out the possibility that one of the justices authored both a signed opinion and the unsigned per curiam. Moreover, this step is useless when there is no separately written opinion.

The article also recommends culling the files of retired Justices like Blackmun and White, which are available in the Manuscript Division of the Library of Congress.[40] Apart from practical access difficulties, the files are incomplete records of communications between the justices and other confidants. Some useful narratives may be pieced together with effort, but these records are still likely to prove incomplete and unsatisfactory. Furthermore, this information only becomes available after a justice retires, which will significantly delay authorship investigations in the vast majority of cases. Even after those files become available, there is no guarantee that the information found therein will be of any use.

Investigations into unsigned opinions from federal appellate and state courts will experience additional problems.[41] Unlike the Supreme Court, these courts are less likely to maintain robust records of any behind-the-scenes happenings. Moreover, such opinions have less practical significance relative to U.S. Supreme Court opinions. In turn, this consideration may imply that historians and other parties have less motivation to investigate how these courts arrived at their decisions and to publish such findings for the benefit of further research.

[38] See Gizzi & Wasby, *supra* note 13, at 116.
[39] *Id.*
[40] *Id.* at 116-17.
[41] *Id.* at 118.

2.2.4 Solving Attributional Questions Algorithmically

Our approach, involving algorithmic natural language analysis, presents several advantages over the aforementioned approaches to determining authorship. First, access is practically a non-issue. Using only public domain opinions of known authorship, we can create a dataset from which we can analyze the natural language of any given opinion. A sufficient quantity of opinions for the dataset is often freely available through resources like the Cornell Legal Information Institute (LII).[42]

We need not access the Library of Congress or put together an incomplete puzzle from the files of retired justices. For our system, we need only an unsigned opinion and a sufficiently large bank of signed opinions from the potential authors. Part V lays out the results from applying our algorithm to every unsigned opinion of the Roberts Court. But for illustrative purposes, the next few sections describe our approach when used on a high-profile test case with an opinion whose authorship was hotly contested.

2.3 Test Case: Obamacare

In June 2012, the Supreme Court issued its ruling in *National Federation of Independent Business v. Sebelius*,[43] which largely upheld the 2010 Patient Protection and Affordable Care Act (PPACA). This highly controversial decision contained what was originally an unsigned dissenting opinion, the authorship of which was a popularly debated issue.

The Sebelius decision was surprising because Chief Justice John Roberts gave the deciding vote, siding with the more liberal justices.[44] The Chief Justice rejected the government's argument that Congress was authorized to enact PPACA's individual insurance coverage mandate under the Commerce Clause, but accepted the govern-

[42] *Supreme Court Collection*, LEGAL INFORMATION INSTITUTE, `http://www.law.cornell.edu/supct/` (last visited Feb. 11, 2013)

[43] 132 S. Ct. 2566 (2011).

[44] See, *e.g.*, John T. Bennett, "Law of the Land: Supreme Court Upholds 'Obamacare,"' US NEWS (June 28, 2012), available at `http://www.usnews.com/news/articles/2012/06/28/law-of-the-land-supreme-court-upholds-obamacare`.

ment's alternative position that the mandate was authorized by Congress's power to enact taxes.[45] Together with the liberal justices, who would have accepted both government arguments, the Chief Justice provided the necessary fifth vote to uphold the law.[46]

Many experts had predicted that (1) the Court would overturn PPACA[47] and (2) the pivotal vote would come from Justice Anthony Kennedy.[48] After the decision, there was speculation that the Chief Justice had switched sides between the time that the case was heard and the time the decision was announced.[49] There was further speculation that the formerly unsigned dissent (later attributed to Justices Kennedy, Scalia, Thomas, and Alito)[50] had originally been a majority opinion authored by Chief Justice Roberts.[51]

The following pieces of evidence have been offered to support this hypothesis. First, the opening section of the joint dissent authored by Kennedy et al. (the "joint dissent") never mentions the Court's majority opinion to uphold the PPACA, written by Chief Justice Roberts.[52] Typically, dissenting and concurring opinions will highlight in the first few paragraphs the reason for authoring a separate opinion, as indeed Justice Ginsburg's opinion[53] and Justice Thomas's opinion[54] do. Instead, the joint dissent only contains arguments against points made by the government attor-

[45] 132 S. Ct. at 2587.

[46] *Id.* at 2594.

[47] See, *e.g.*, Peter Ferrara, "Why the Supreme Court Will Strike Down All of Obamacare," FORBES (April 5, 20120), *available at* http://www.forbes.com/sites/peterferrara/2012/04/05/why-the-supreme-court-will-strike-down-all-of-obamacare/.

[48] See, *e.g.*, Peter J. Boyer, "Reading Justice Anthony Kennedy's Leanings on Obamacare," THE DAILY BEAST (Apr. 2, 2012), http://www.thedailybeast.com/newsweek/2012/04/01/reading-justice-anthony-kennedy-s-leanings-on-obamacare.html.

[49] See, *e.g.*, Sabrina Siddiqui, "John Roberts' Switch on Obamacare Sparks Fascination with Supreme Court, Possible Leaks," HUFFINGTON POST (July 3, 2012), http://www.huffingtonpost.com/2012/07/02/justice-roberts-obamacare-supreme-court-leaks_n_1644864.html; Paul Campos, "Did John Roberts switch his vote?," SALON.COM (June 28, 2012), http://www.salon.com/2012/06/28/did_john_roberts_switch_his_vote/.

[50] 132 S. Ct. 2642.

[51] See, *e.g.*, Avik Roy, "The Inside Story on How Roberts Changed His Supreme Court Vote on Obamacare," FORBES (July 1, 2012), available at http://www.forbes.com/sites/aroy/2012/07/01/the-supreme-courts-john-roberts-changed-his-obamacare-vote-in-may/.

[52] See 132 S. Ct. 2642-44.

[53] *Id.* at 2602 (Ginsburg, J. concurring in part and dissenting in part).

[54] *Id.* at 2677 (Thomas, J. dissenting).

neys defending PPACA and Justice Ginsburg's opinion. In this respect, the joint dissent reads more like a majority opinion (with a corresponding dissent by Justice Ginsburg), rather than a dissent arguing against Chief Justice Robert's opinion.

Second, whereas a typical majority opinion will refer to the decision-maker as the Court and describe the majority justices using the collective pronoun "we," indicating solidarity, dissents and concurrences typically refer to themselves individually using less inclusive pronouns like "I." True to form, Chief Justice Roberts' majority opinion follows this pattern, as does Justice Ginsburg's opinion (which is joined by Justice Sotomayor),[55] and Justice Thomas's opinion.[56] The joint dissent, however, does not follow the pattern.[57]

Third, although there are two dissenting opinions in this case — the joint dissent and another authored by Justice Thomas alone — the joint dissent refers to Justice Ginsburg's opinion concurring in part and dissenting in part in the following way: "*The* dissent claims that we 'fai[l] to explain why the individual mandate threatens our constitutional order.' Ante, at 2627. But we have done so."[58] It is peculiar that this joint dissent does not acknowledge itself as one of the two opinions dissenting in full. It is even more peculiar that the joint dissent calls Justice Ginsburg's opinion "the dissent" when her opinion is, in fact, only partially dissenting. As a practical matter, it would appear that either the joint dissent or Justice Thomas' dissent is more deserving of being dubbed "the dissent." This sentence would make more sense as a majority opinion critiquing a sole dissenting opinion (on the assumption that, in this counterfactual scenario, Justice Thomas would have joined the counterfactual majority or at least changed his dissent to a concurrence).

Fourth, Justice Ginsburg provided the following criticism of Chief Justice Roberts's

[55]Id. at 2609 (Ginsburg, J. concurring in part and dissenting in part) ("*I* agree with... *I* therefore join... [H]owever, *I* would hold... (emphasis added)). Note also that Justice Ginsburg uses first person singular pronouns despite the fact that Justices Sotomayor, Breyer, and Kagan joined in all or part of the Justice Ginsburg's opinion.

[56]*Id.* at 2677 (Thomas, J. dissenting) ("*I* dissent... *I* write separately to... *I* adhere to *my* view" (emphasis added)).

[57]*Id.* at 2432 (Scalia, Kennedy, Thomas, Alito, Js. dissenting) (emphasis added) ("*We* conclude... .").

[58]Id. at 2659 (Scalia, Kennedy, Thomas, Alito, Js. dissenting).

reasoning in the majority opinion:

> In failing to explain why the individual mandate threatens our constitutional order, THE CHIEF JUSTICE disserves future courts. How is a judge to decide, when ruling on the constitutionality of a federal statute, whether Congress employed an independent power, ante, at 2591, or merely a derivative one, *ante*, at 2592. Whether the power used is substantive, *ante*, at 2592, or just incidental, ante, at 2592? The instruction THE CHIEF JUSTICE, in effect, provides lower courts: You will know it when you see it.[59]

There is a direct response to this argument, but it appears in the joint dissent, not Chief Justice Roberts' majority opinion.[60]

An alternate hypothesis is that the joint dissent was actually written mostly by Justices Kennedy and Scalia, as argued by a detailed news article with sources allegedly close to the Supreme Court.[61] This article also gives an explanation as to why the joint dissent does not engage Justice Roberts' majority opinion:

> The majority decisions were due on June 1, and the dissenters set about writing a response, due on June 15. The sources say they divided up parts of the opinion, with Kennedy and Scalia doing the bulk of the writing. The two sources say suggestions that parts of the dissent were originally Roberts' actual majority decision for the court are inaccurate, and that the dissent was a true joint effort.
>
> The fact that the joint dissent doesn't mention Roberts' majority was not a sign of sloppiness, the sources said, but instead was a signal the conservatives no longer wished to engage in debate with him.[62]

[59] *Id.* at 2627-28 (Ginsburg, J. concurring in part and dissenting in part).

[60] Id. at 2649 (Scalia, Kennedy, Thomas, Alito, Js. dissenting).

[61] Jan Crawford, "Roberts switched views to uphold health care law," CBS NEWS, (July 1, 2012) available at `http://www.cbsnews.com/8301-3460_162-57464549/roberts-switched-views-to-uphold-health-care-law/`.

[62] *Id.*

A further interview with Justice Ginsburg suggests that she wrote her own dissent early on, believing that the Chief Justice would strike down the individual mandate:

> Ginsburg quickly began drafting the dissenting statement on that issue, portions of which she read from the bench on the day the ruling was announced. "I had a draft of the dissent before the chief circulated his opinion because I knew it would be impossible to do" as the term went into the final month of June and several cases culminated.[63]

Ultimately, the evidence is mixed as to whether both the majority opinion and the joint dissent were authored by the Chief Justice. Applying authorship attribution techniques, we aspire to determine quantitatively which of these hypotheses is more plausible. Our model assigns the highest probability to Justices Scalia and Kennedy, not Chief Justice Roberts, as the author of the dissenting opinion, which supports the "Crawford" theory of authorship.[64]

2.4 Experimental Setup

2.4.1 Experimental Questions

To solve this attribution question, we focused on whether features of each justice's writing styles could be used to predict which justice authored which opinion. The specific questions that we sought to answer through our experiments were the following:

1. Can statistical authorship attribution methods accurately predict which of the Supreme Court justices authored a given opinion?

2. What are the words and stylistic features that most distinguish different Supreme Court justices, and what do they reveal about the writing styles of different justices?

[63] Joan Biskupic, "Exclusive: Justice Ginsburg shrugs off rib injury," REUTERS (Aug. 8, 2012) available at `http://www.reuters.com/article/2012/08/09/us-usa-court-ginsburg-idUSBRE87801920120809`.

[64] *See* Crawford, *supra* note 62.

3. Which author(s) does the model predict for the majority and dissenting opinions written in *Sebelius*?

Our work is part of the growing literature on applying algorithmic natural language processing tools to legal opinions. Recent work has explored the evolution of language in Supreme Court texts over time,[65] the conversational dynamics of Supreme Court oral arguments,[66] and the role of law clerks in the opinion-writing process.[67] Our work applies an analogous quantitative approach to investigate the authorship of unsigned opinions and opinions of controversial attribution, leveraging advances in computational power and machine learning algorithms to infer authorship with high accuracy.

2.4.2 Data Preparation

We obtained texts of Supreme Court opinions from the Cornell Legal Information Institute (LII).[68] From this source, we downloaded all Supreme Court decisions written by the nine justices currently sitting on the Court who have served during the tenure of Chief Justice John Roberts, i.e., from 2005 to 2011. For each case, we extracted the majority and minority opinions if they existed, keeping track of their respective authors. We masked the surnames of the justices themselves and years, to avoid simply using name or year information to identify the author. Concurrences to either the majority or minority opinion were not included in our corpus of possible cases, as our focus was on predicting authorship of majority and dissenting opinions.

Using these criteria, our dataset consists of 568 opinions. In addition to these 568 opinions, we obtained the majority (signed by Chief Justice Roberts) and minority (signed by Justices Scalia, Kennedy, Thomas, and Alito) opinions of the Sebelius

[65] Daniel M. Katz et al., *Legal n-grams? a simple approach to track the evolution of legal language*, in PROCEEDINGS OF THE 24TH INTERNATIONAL CONFERENCE ON LEGAL KNOWLEDGE AND INFORMATION (2011).

[66] Timothy Hawes et al., *Elements of a computational model for multi-party discourse: The turn-taking behavior of Supreme Court justices*, 60(8) J. AM. SOC'Y INFO. SCIENCE & TECH. 1607 (2009).

[67] Jeffrey S. Rosenthal & Albert H. Yoon, *Detecting multiple authorship of united states supreme court legal decisions using function words*, 5(1) ANNALS OF APPLIED STATISTICS 283 (2011).

[68] See *supra* note 42.

decision and 65 per curiam decisions by the Roberts court (until November 2012). We obtained these 67 opinions using the same download protocol. Our objective is to make meaningful authorship predictions for these 67 opinions.

2.4.3 Machine Learning System Overview

Our machine learning approach follows the paradigm of "supervised learning": our algorithms identify characteristics (called "features") of each justice's writing style from opinions known to be authored by him or her. These characteristics are encoded in a statistical prediction model that describes the writing styles of the justices under consideration. Given a new opinion with an unknown author, the model predicts which justice wrote the opinion. It is worth noting that "supervised learning-based authorship attribution" has been applied to a wide range of literary, historical, and contemporary domains,[69] including studies on the Federalist Papers,[70] Shakespeare's plays,[71] and more recently, on large numbers of authors in online blogs or forums.[72]

Our system builds upon some early work in judicial authorship, but with a greater focus on predicting who authored a particular opinion. In the aforementioned studies on the role of clerks in opinion-writing, researchers found that they could differentiate between pairs of Supreme Court justices using just 63 function words.[73] Our work leverages a much larger number of words and word phrases (about 100 times as many) culled from the opinions themselves, a process that is feasible and inexpensive on today's computers. In doing so, we are able to handle the problem of accurately

[69]For a detailed review of classification techniques, useful features, and application areas, see Moshe Koppel et al., *Computational methods in authorship attribution*, 60(1) J. AM. SOC'Y INFO. SCIENCE & TECH. 9 (2009), *available at* `http://dx.doi.org/10.1002/asi.v60:1`; Efstathios Stamatatos, *A survey of modern authorship attribution methods*, 60(3) J. AM. SOC'Y INFO. SCIENCE & TECH. 538 (2009), available at `http://dx.doi.org/10.1002/asi.21001`.

[70]Frederick Mosteller & David Wallace, INFERENCE AND DISPUTED AUTHORSHIP: THE FEDERALIST (1964).

[71]Thomas V.N. Merriam & Robert A.J. Matthews, *Neural computation in stylometry ii: An application to the works of Shakespeare and Marlowe*, 9(1) LITERARY & LINGUISTIC COMPUTING 1 (1994).

[72]Moshe Koppel et al., *Authorship attribution with thousands of candidate authors*, *in* PROCEEDINGS OF THE 29TH ANNUAL INTERNATIONAL ACM SIGIR CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL 659 (2006).

[73]See Rosenthal & Yoon, *supra* note 68.

predicting which of the nine justices wrote an unsigned or controversial opinion.

Within this framework, our corpus of opinions is divided into three datasets:

1. "Training set": Of the 568 signed opinions, we take 451 of them (80%) to "train" the authorship attribution system. The machine learning algorithms, described further below, are given both the text and the author of each of these opinions and learn the parameters of this system from this training data.

2. "Validation set": The remaining 117 (20%) signed opinions are deliberately excluded from the training process, and the authorship attribution system is used to "predict" the authors of these cases. The trained system is given only the text of these opinions and asked to provide an authorship prediction. Given that the author is known in these cases, the prediction has little scholarly value in itself; however, the performance of the system on these 117 cases, which can be measured by comparing the prediction to the actual author, provides some indication of the system's predictive capabilities.

3. "Test set": The opinions of the Sebelius decision and the 65 per curiam opinions of the Roberts Court form the final test set. Similar to the validation set, these cases are excluded from the training process. The results of this analysis are shown in Parts IV and V.

For our Supreme Court authorship attribution task, there are nine justices and thus the model must accurately choose among nine choices for each opinion. Along with high classification accuracy, we identified the following desiderata (in consultation with a practicing attorney familiar with Supreme Court cases and customs) for our classification scheme: (1) the features should be intuitive and easy to understand; (2) the prediction should have a confidence score for the correctness of the predicted justice; (3) the prediction should produce a meaningful probability distribution over the nine justices; and (4) the predicted author should be the justice with the highest probability.

2.4.4 Design of Authorship Attribution System

Building a statistical authorship attribution model requires three main design decisions: (1) how to represent each judicial opinion, (2) which statistical machine learning model to use, and (3) how to select features. This section describes how we made these decisions, guided by established practices in constructing authorship attribution systems, quantitative experiments (detailed in the next section) to validate our choices, and the specific questions we sought to answer with this model.

Document Representation. To build the statistical authorship attribution model, the opinions must be characterized in terms of numerical features. Human experts might examine each justice's vocabulary richness, grammatical patterns, opinion length, or other writing style characteristics to try to distinguish between them. In our method, we use straightforward features that serve as a proxy for these intuitions: the presence of one-, two-, and three-word sequences (known in natural language processing as unigrams, bigrams, and trigrams, respectively; "n-grams" is the term for any sequence of n words) in a document. Table 2.1 illustrates how a single sentence from the majority opinion of the Sebelius decision can be described by these word sequences. Applied over an entire written opinion, such features encode information about vocabulary, syntax (such as the use of "however" in the middle of sentence), and subject matter. We do not eliminate capitalization or punctuation marks, which may also be indicative of writing style; for example, "Namely" is a different feature than "namely.". In addition, we did not discard punctuation immediately following words because these characteristics might differentiate the justices' writing styles. These n-gram features have been effective in a wide range of authorship attribution efforts.[74]

Experimental Evaluation: We evaluated the authorship attribution model using four different sets of features: (1) unigrams only; (2) bigrams only; (3) trigrams only; and (4) the combination of unigrams, bigrams, and trigrams. A comparison of model performance across these sets of features allows us to measure the incremental value of longer sequences of words. The results are presented in Table 3 in the next section.

[74]See Koppel et al., *supra* note 70.

Table 2.1: Example of sentence decomposed into unigrams, bigrams, and trigrams

Full sentence	It does not, however, control whether an exaction is within Congress's power to tax.
Unigrams	"It"; "does"; "not,"; "however,"; "control"; "whether"; "an", "exaction", "is", "within"; "Congress's"; "power"; "to" "tax."
Bigrams	"It does"; "does not,"; "not, however,"; "however, control"; "control whether"; "whether an"; "an exaction"; "exaction is"; "is within"; "within Congress's"; "Congress's power"; "power to", "to tax."
Trigrams	"It does not"; "does not, however"; "not, however, control"; "however, control whether"; "control whether an"; "whether an exaction"; "an exaction is"; "exaction is within"; "is within Congress's"; "within Congress's power"; "Congress's power to"; "power to tax."

Model Selection. Given our goal of producing a meaningful probability distribution of authorship, we trained a maximum entropy (MaxEnt) statistical model — which enjoys widespread use in text classification tasks — for our authorship attribution system.[75] Specifically, we designed our model to compute the following probability:

$$P(y|x_i) = \frac{\exp(\theta \cdot \phi(x, y_i))}{\sum_{k=1}^{9} \exp \theta \cdot \phi(x, y_k)} \tag{2.1}$$

where:

- x: Input text opinion
- y_i: Dependent variable representing justice i, where i ranges from 1 to 9 (corresponding to the nine serving justices)
- $P(y_i \mid x)$: Probability of justice y_i as author, given input opinion x
- $\phi(y_i \mid x)$: Feature vector with entries corresponding to each of the n-gram

[75] Adam L. Berger et. al, *A maximum entropy approach to natural language processing*, 22(1) COMPUTATIONAL LINGUISTICS 39 (1996).

features for each justice.

- θ: Weight vector (the set of coefficients on each feature)

For a given document, the MaxEnt model computes a score for each justice, i, that is a weighted sum of the n-gram features. Using the training data, the machine learning algorithms automatically learn the parameters of the weight vector to maximize the likelihood of the training data; that is, the algorithm adjusts the weights to best "explain" the data. The form of the MaxEnt model ensures that $P(y_i|x)$ is between 0 and 1 and that these probability values sum to 1, meaning that the output of the model can be interpreted as a probability distribution. This relatively simple approach has been used successfully in other text classification problems.[76] We used Apache OpenNLP[77] for the MaxEnt model and WEKA[78] , two open-source machine learning software packages, for the baseline methods outlined below.

In summary, the authorship attribution system computes a probability distribution over the nine justices for a given written opinion. The justice with the highest probability is used as the predicted author, as discussed in the next section.

Experimental Evaluation. To validate our choice of the MaxEnt authorship attribution model, we compared it to other common machine learning algorithms using the same set of features. These other common machine learning algorithms are:

1. Decision trees (DT)[79]: Instead of taking a weighted sum of features, decision tree models learn deterministic rules directly from the feature set. For example, the decision tree may use the presence or absence of a particular n-gram to predict one justice as opposed to another. These conceptually simple models may suffer in performance because certain features might be indicative, but not determinative, of certain authors. In probabilistic models, negative evidence

[76] Adwait Ratnaparkhi, *Maximum entropy models for natural language ambiguity resolution*, Ph.D. thesis, University of Pennsylvania (1998).

[77] "The OpenNLP library is a machine learning based toolkit for the processing of natural language text." OpenNLP is available at `http://opennlp.apache.org/`.

[78] "[WEKA] is a collection of machine learning algorithms for data mining tasks," and is available under a GNU General Public License at `http://www.cs.waikato.ac.nz/ml/weka/`.

[79] *See* CHRISTOPHER M. BISHOP, PATTERN RECOGNITION AND MACHINE LEARNING, Section 16.4 (2006).

against a particular author can be outweighed by positive evidence in favor of the author.

2. Naive Bayes (NB) classification[80]: By assuming that features are statistically independent, the Naive Bayes model calculates a probability that a justice wrote an opinion by multiplying the conditional probabilities of each feature given the justice. In other words, each feature's "contribution" to the model is computed separately because it assumes that all of the features are statistically independent. As a result, the NB model is substantially simpler and faster to train than the MaxEnt model, the latter of which involves learning the weights for all of the features. However, the NB model may not perform as well because it does not attempt to search through all possible weights to maximize performance.

3. Pairwise-coupled support vector machines (SVM)[81]: Some authorship attribution applications have reported state-of-the-art results with support vector machines, which also learn weights on features using a different mathematical formulation.[82] We used pairwise-coupled SVMs, in which the opinions of each possible pair of justices are trained. The output of each classifier is a "vote" for one justice over another, and the justice with the most overall "votes" is the predicted author. One challenge related to our desiderata is that the votes may be difficult to interpret as meaningful probabilities.

Table 3 in the next section compares the results of each of these three models with the MaxEnt model.

Feature Selection. We also decided to selectively limit which n-gram features were used in the authorship attribution model. Given the length and quantity of opinions, there are hundreds of thousands of possible n-grams in the set of Supreme

[80] *See* DANIEL JURAFSKY & JAMES H. MARTIN, SPEECH AND LANGUAGE PROCESSING: AN INTRODUCTION TO NATURAL LANGUAGE PROCESSING, COMPUTATIONAL LINGUISTICS, AND SPEECH RECOGNITION, Section 20.2.2 (2009).

[81] BISHOP, *supra* note 80, at ch.7.

[82] For a description of the SVM implementation we used, see John C. Platt, *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*, in ADVANCES IN KERNEL METHODS: SUPPORT VECTOR LEARNING, (Bernhard Schoelkopf et al., eds., 1998).

Court opinions, but not all of them are likely to be useful. For instance, an n-gram could reflect vocabulary specific to a single case; associating it with a particular author would not be useful for predicting authorship in the validation set or in cases where the author is unknown. In addition, having too many features in our model could make it prone to "over-fitting" — the results would not generalize to opinions in our validation set.[83]

First, we considered only features that appear in a minimum of 20 opinions in our training set, thereby eliminating very case-specific or rare language. Then, we considered two feature selection methods that are commonly used for text processing: document frequency and information gain.[84] In both cases, we computed a score for each eligible n-gram feature and then took the highest-ranked features:

1. Document Frequency (DF) computes the frequency score of the n-gram simply by counting the number of documents that include the n-gram. We selected the 3000 most frequent unigrams, 1000 most frequent bigrams, and 1000 most frequent trigrams in the training set for our DF-based model. While DF-based feature selection is simple, it has been shown empirically to be as effective as more sophisticated methods in text classification tasks.[85] Choosing frequent n-grams could be a reasonable feature selection method because n-grams that reflect common writing styles or patterns are likely to appear in the opinions in our validation set.

2. Instead of scoring each n-gram feature by its frequency of appearance, Information Gain (IG) measures the contribution of a particular feature to differentiating among the justices, which is the ultimate goal of our authorship attribution model. Specifically, we compute the weighted average entropy of each feature, f:

[83] BISHOP, *supra* note 80, at ch. 1.

[84] Yiming Yang and Jan O. Pedersen, *A Comparative Study on Feature Selection in Text Categorization*, in PROCEEDINGS OF THE 14TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING (1997).

[85] *Id.*

$$IG(f) = P(f_{\text{present}}) \sum_{k=1}^{9} P(j_k \mid f_{\text{present}}) \log P(j_k \mid f_{\text{present}}) + P(f_{\text{absent}}) \sum_{k=1}^{9} P(j_k \mid f_{\text{absent}}) \log P(j_k \mid f_{\text{absent}}) \quad (2.2)$$

where:

- $P(f_{\text{present}})$: The fraction of documents that contain the n-gram feature.
- $P(f_{\text{absent}})$: The fraction of documents that do not contain the n-gram feature.
- $\log P(j_k \mid f_{\text{present}})$: A measure of the non-uniformity of the probability distribution of opinions authored by different justices, conditioned on the presence of the feature f. This definition is precisely the negative of the entropy of the probability distribution[86] — the more non-uniform the probability distribution, the higher the score.
- $\log P(j_k \mid f_{\text{absent}})$: A measure of the non-uniformity of the probability distribution of opinions by justice, conditioned on the absence of feature f.

Once we computed these scores for every eligible n-gram, we chose the n-grams with the highest scores. For the IG model, we chose 3002 unigrams (similar to the number of unigrams in the DF model). We included bigrams and trigrams that had an IG score at least as high as one of the included unigrams. As a result, we selected 2714 bigrams and 746 trigrams into our model.

Table 2.2 shows examples of n-gram features that were selected using DF, IG, or both feature selection methods. Some of the features selected using DF, such as "the" or "at", had low IG scores because they appear in almost every document; consequently, they were not chosen as features when evaluated for information gain. Despite substantial differences in the two feature sets, it is worth noting that many

[86] *See* Claude Shannon, *Prediction and entropy of printed English*, 30(1) BELL SYSTEMS TECHNICAL JOURNAL 50 (1951).

Table 2.2: Examples of n-gram features selected by Document Frequency (DF) and Information Gain (IG) methods

Features only in DF model (3154 features)	**Features in DF and IG models (1846 features)**	**Features only in IG model (4616 features)**
"the", "at", "exclude", "conceded", "refers to", "entitled to", "exception to the", "see no reason"	"stated,", "consequently", "declared", "assumption", "Even if", "consideration of", "and the case", "fact that the"	"Furthermore,", "troubling", "undisturbed", "evidently", "That would", "assert that", "the premise that", "is apparent that"

of the types of n-grams are quite similar — they seem to encode characteristics of justices' writing styles. A reasonable hypothesis is that a model with features selected under our information gain method is more likely to yield better results, but this must be validated experimentally.

Experimental Evaluation. Given that both feature selection methods are commonly used in algorithmic text analysis, we present results using both DF and IG for the Sebelius decision. Showing the findings from both models could give greater insight into authorship in this decision.

2.5 Empirical Results and Discussion

2.5.1 Feature Sets and Classification Methods

Table 3 summarizes the performance of feature types and classification models mentioned in the previous section. For consistency in this set of experiments, the DF method of selecting features was used for all of these models, meaning that the feature set is the same. These results were obtained by ten-fold cross-validation, in which a model is trained on 90% of the training data, the results are computed for the remaining 10%, and the 90/10 split is repeated a total of ten times to judge the performance on the entire, 451-opinion training set. The steps of cross validation

Table 2.3: Authorship prediction accuracy by feature set and classifier

Features	DT	NB	SVM	MaxEnt
Unigrams	0.322	0.514	0.734	0.736
Bigrams	0.266	0.443	0.559	0.588
Trigrams	0.244	0.459	0.501	0.548
Uni/Bi/Trigrams	0.301	0.527	0.747	0.752

were as follows:

1. We randomly divided the set of documents into ten equal partitions.

2. .We "held out" one partition and trained the authorship attribution model on the remaining nine partitions. Then, we tested the trained model on the "held out" partition.

3. We repeated step 2 once for each of the ten partitions, then combined the results. Given that there were ten partitions, this process is called ten-fold cross-validation.

It appears that the scores for correctly and incorrectly classified opinions are drawn from different, albeit overlapping, distributions for the MaxEnt-DF and MaxEnt-IG models. In general, for both models, higher output scores correspond to higher likelihood that the prediction is correct.

As seen in Table 2.3, the unigram features provide good prediction accuracy on their own, with some improvement when bigrams and trigrams are added as features. The maximum entropy (MaxEnt) and support vector machine (SVM) classifiers outperform the Naive Bayes (NB) and decision tree (DT) models. Additionally, the MaxEnt model performs slightly better than the SVM and has the added property of probability distributions over the justices. Overall, these findings help justify the use of unigrams, bigrams, and trigrams as features in a MaxEnt framework in our authorship attribution model. The remaining results and analysis in this section focus on variants of the MaxEnt model exclusively.

Table 2.4: Performance of MaxEnt models with Document Frequency (DF) and Information Gain (IG) feature selection

	MaxEnt-DF	MaxEnt-IG
Accuracy on test set	0.761 (89/117)	0.812 (95/117)

2.5.2 Comparison of Feature Selection Models

Given the strong performance and desirable properties of the MaxEnt classification model, we evaluated the model on our 117-opinion validation set using the document frequency (MaxEnt-DF) and information gain (MaxEnt-IG) feature selection methods. We obtained the features and model weights from the 451-opinion training set. Table 2.4 compares the performance of the MaxEnt model using the document frequency (MaxEnt-DF) and information gain (MaxEnt-IG) methods of feature selection. IG results in a performance of 81.2% (95 out of 117 cases correct), while DF has a lower accuracy at 76.1% (89 out of 117). The somewhat lower performance of MaxEnt-DF is likely because some features do not help to differentiate the justices and may merely add noise to the model.

2.5.3 Interpreting Authorship Attribution Model Scores

In addition to predicting a justice, the MaxEnt models also provide an output score for each justice that can be interpreted as a probability. Figure 2-1 shows the distribution of maximum scores for correctly and incorrectly classified opinions in the training set. In order to characterize the behavior of the model scores on our entire set of opinions, we obtained these results through the machine learning technique of ten-fold cross-validation, similar to the approach used to compare the different learning models.

Instead of simply taking the justice with the highest probability as the prediction, a more refined prediction system that "abstains" (makes no prediction) below a certain output probability can be constructed. The ratio of correctly predicted cases to incorrectly predicted cases increases as this threshold increases at the expense of a larger number of abstentions. This result is visualized in Figure 2-2 and provides

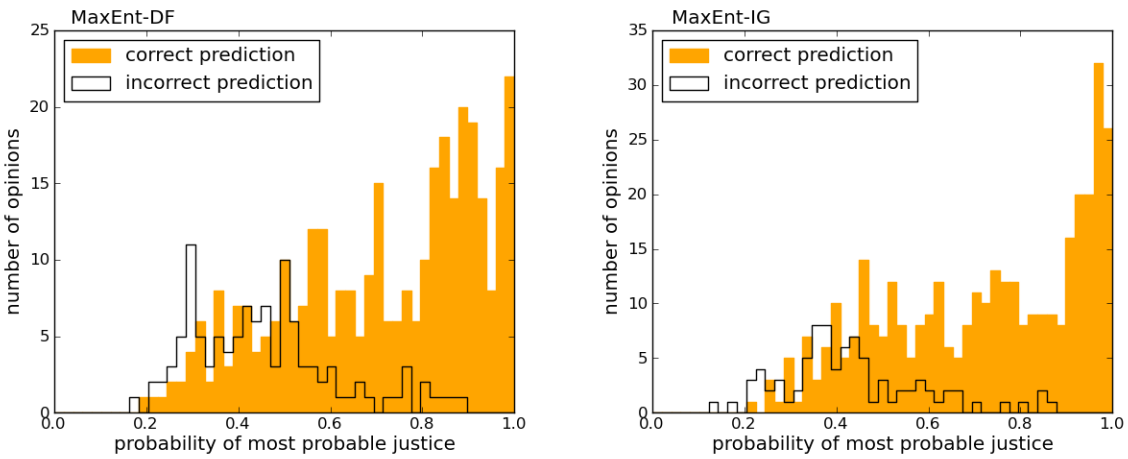


Figure 2-1: Histograms of probabilities of most probable justice for MaxEnt-DF (left) and MaxEnt-IG (right) models.

another illustration of the differences between the MaxEnt-DF and MaxEnt-IG models. For example, in the MaxEnt-IG model, a threshold probability of 0.43 results in 90% prediction accuracy, with abstentions on just 19.3% of all cases. In contrast, to achieve 90% prediction accuracy, the MaxEnt-DF model must set a threshold of 0.52, abstaining on 35.1% of all cases. Overall, these plots illustrate the probabilistic nature of our authorship attribution model. Based on the desired application of the model, one could set different abstaining thresholds, depending on the level of confidence desired.

2.5.4 Insights on Writing Styles

To provide some insight into how our authorship attribution system predicts which justice wrote an opinion, Table 2.5 shows some of the most predictive unigrams, bigrams, and trigrams for each justice from the MaxEnt-IG model. This table was computed by determining n-grams that appear disproportionately more often for each justice. Some noteworthy insights include:

1. The highly predictive n-grams are largely topic-invariant function terms; that is, the informative features more frequently reflect the writing style of the justice as opposed to specific subjects. For example, the term "consequently" appears

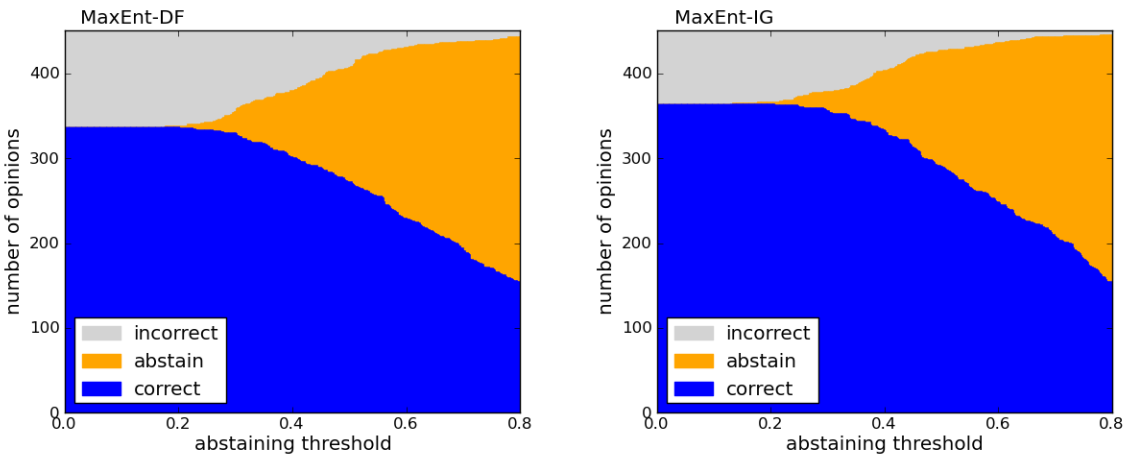


Figure 2-2: Effect of abstaining threshold on size of correct, incorrect, and abstaining classes of opinions for MaxEnt-DF (left) and MaxEnt-IG models (right).

in 99 different opinions in our training dataset; Justice Breyer wrote 79 of these opinions.

2. Some predictive n-grams begin with capitalized words, including "For one thing" (Breyer), "Notably," (Ginsburg), and "The question is" (Kennedy). These correspond to words at the beginning of sentences, indicating that how different justices start sentences provides clues about authorship.

3. Some predictive n-grams include punctuation like commas and periods, including "reason stated, the" (Ginsburg), "the first place." (Roberts)", and "foregoing reasons," (Thomas). By not eliminating punctuation from the text, the authorship attribution model is able to leverage these stylistic features.

2.5.5 Controlling for Clerks

The training/test split in the experiments above was random with respect to the year in which the opinion was written, meaning that, at least to some extent, a justice's writing style in a given year can be predicted from his or her writings in other years. Our model does not explicitly consider the role of law clerks, who typically serve year-long terms, in the writing process; rather, it assumes that the features of a justice's writing are similar from year to year.

Table 2.5: Informative features by justice

Justice	Unigrams	Bigrams	Trigrams
Alito	"fundamentally", "widely", "regarded"	"set out", "noted above", "is generally"	"set out in", "and we have", "the decision of"
Breyer	"consequently", "Hence", "thing,"	"can find", "wrote that", "For one"	"in respect to", "For one thing", "That is because"
Ginsburg	"Notably,", "observed,", "stated,"	"reasons stated,", "stated, the", "case concerns"	"stated, the judgment", "reasons stated, the"
Kagan	"enables", "earlier," "matters."	"result is", "after all,", "the theory"	"do not think", "Court has never", "even when the"
Kennedy	"however.", "responsibilities", "Though"	"It held", "so the", "or she"	"The question is", "as a general", "he or she"
Roberts	"pertinent", "accordingly", "Here"	"first place.", "only be", "given that"	"the first place.", "without regard to", "a general matter,"
Scalia	"utterly", "thinks", "finally"	"Of course", "since it", "is entirely"	"That is not", "the present case", "is hard to"
Sotomayor	"observes", "lawsuits", "heightened"	"Committee on", "federal and", "correct that"	"circumstances in which", "see also, Brief", "federal and state"
Thomas	"Therefore," "However,", "explaining"	"address whether", "foregoing reasons," "Court holds"	"hold that it", "For the foregoing", "the foregoing reasons"

Table 2.6: Prediction accuracy of models trained on opinions from different years

Partition	**Accuracy**
Year-based	74.8%
Random	78.9%

To test this assumption, we once again applied the cross-validation technique (similar to how we studied the model confidence scores) and divided the set of documents in two different ways. Specifically, we took signed opinions from the Roberts Court in the 2005-2006, 2006-2007, 2007-2008, 2008-2009, 2009-2010, and 2010-2011 sessions and compared the performance of our model in two ways:

1. For each of the six annual sessions, we trained the MaxEnt-IG model on the other five sessions and validated on cases from the omitted session.

2. The signed opinions were randomly divided into six partitions, irrespective of the year. For each partition, we trained the MaxEnt-IG model on the other five partitions and validated on opinions from the omitted partition.

In both cases, we aggregated the results from each of the six runs, as shown in Table 6. Evidently, training on other years has only a slight adverse impact on the accuracy of the model. This may suggest that a different set of clerks have some impact on a justice's writing style, although other factors, such as a justice's own drift in writing style, may contribute to this result. Overall, though, the higher performance of the randomized model suggests that the combination of training data from other years and the same year works well for predicting authorship.

2.5.6 Authorship Prediction for *Sebelius*

To infer authorship in the Sebelius decision, we trained the MaxEnt-DF and MaxEnt-IG models on the 568 cases in our dataset and ran them on the majority opinion signed by Chief Justice Roberts and the joint dissent. Figure 2-3 and Figure 2-4

illustrate the resulting probability distributions of the two models. Both MaxEnt-DF and MaxEnt-IG strongly predict Chief Justice Roberts for the majority opinion. For the minority opinion, the MaxEnt-DF model states that Justice Kennedy is the predicted author, but the probability distribution is not as peaked — Justice Scalia has the second-highest probability. Meanwhile, the MaxEnt-IG model is much more confident in Justice Scalia as the author of the joint dissent. Overall, these findings are sensible: Chief Justice Roberts signed the majority opinion, while Kennedy and Scalia are listed as authors of the joint dissent. Both models support the hypothesis that Kennedy and Scalia were authors and prime actors in writing the dissent, and refute the hypothesis that Roberts authored both opinions. The output of the model in the Sebelius decision is an example of the non-partisan, quantitative analysis that the authorship attribution system can provide.

2.5.7 Comparison to Predictions by Domain Experts

To gauge the performance of our authorship attribution system to expert human judgment, we received input from 11 individuals close to the Supreme Court (as past law clerks or lawyers who have argued at least one case before the Supreme Court).[87] We asked each of the respondents to provide their best, informed guess of the authorship of the joint dissent, annotated, if possible, by section. Out of the 11 responses, there were nine who concluded that Kennedy participated in some way, nine for Roberts, six for Scalia, and three for Alito; no other justices were mentioned. The predictions of our model on this case generally seem to be in agreement with these domain experts, although it is worth noting that the MaxEnt-IG model is more confident in Scalia than the domain experts. A summary of the responses from each of the respondents is listed in Table 3.

[87] All of the respondents asked to remain anonymous.

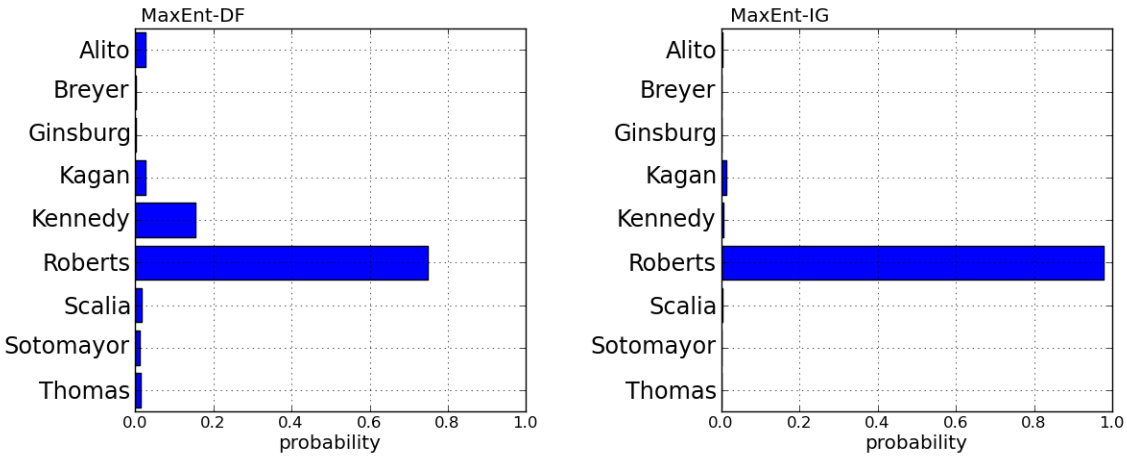


Figure 2-3: Authorship attribution model prediction for Sebelius majority opinion by MaxEnt-DF (left) and MaxEnt-IG (right).

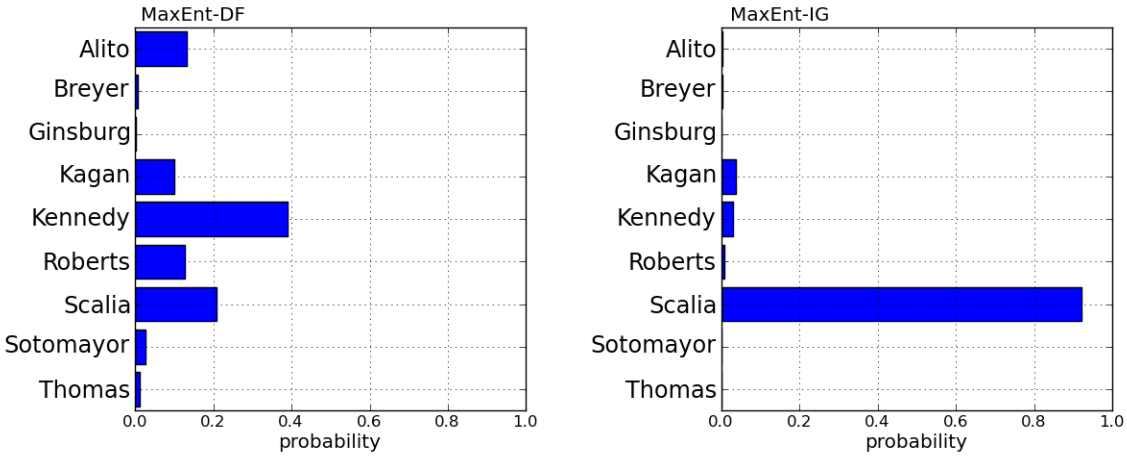


Figure 2-4: Authorship attribution model prediction for Sebelius joint dissent by MaxEnt-DF (left) and MaxEnt-IG (right).

Table 2.7: Predictions of authorship of minority opinion by domain experts

Respondent	Predicted authors (ranked in order of level of contribution)
1	Kennedy, Roberts
2	Scalia, Roberts
3	Roberts, Kennedy
4	Roberts, Scalia, Kennedy, and Alito
5	Kennedy, Alito
6	Scalia, Kennedy, Roberts
7	Scalia, working loosely from a draft by Roberts
8	Kennedy, working loosely from a draft by Roberts
9	Roberts, Kennedy, Scalia
10	Kennedy, Alito, Roberts
11	Scalia, Kennedy

2.5.8 Section-by-Section Analysis

Given that some respondents provided predictions by section, we tested our MaxEnt models on the joint dissent divided into nine sections. We emphasize that the models were not trained on portions of opinions; however, understanding the contributions of different justices to the constituent parts of an opinion could be useful. Sections are frequently delineated according to a precise legal issue and, theoretically, one justice could contribute his or her treatment of a specific legal issue to be incorporated into an opinion written by another justice.

The top predictions and predictions for the MaxEnt-DF and MaxEnt-IG models for each of Sebelius's sections are shown in Table 2.8. The results are somewhat noisy — our respondents did not predict Breyer or Thomas as authors of the joint dissent, and the two models did not agree on every section. However, consistent with its prediction for the entire opinion, the MaxEnt-IG model predicted Scalia for most of the sections. Additionally, none of the predictions suggest that Roberts is the top author, which may lend further evidence against the theory that Roberts authored the minority opinion.

Table 2.8: Prediction of authorship of minority opinion by section

	Predicted author	
Section	MaxEnt-DF	MaxEnt-IG
Introduction	Scalia (0.841)	Scalia (0.544)
Sec. 1 Introduction	Scalia (0.365)	Scalia (0.235)
Sec. 1A	Breyer (0.406)	Scalia (0.284)
Sec. 1B	Scalia (0.730)	Scalia (0.613)
Sec. 1C	Kennedy (0.904)	Scalia (0.590)
Sec. 2	Scalia (0.552)	Scalia (0.891)
Sec. 3	Thomas (0.541)	Scalia (0.344)
Sec. 4	Scalia (0.770)	Alito (0.283)
Sec. 5	Alito (0.385)	Kennedy (0.738)

2.6 Authorship Predictions for Per Curiam Opinions of the Roberts Court

Finally, we tested the MaxEnt-IG model on 65 per curiam opinions of the Roberts Court since 2005, with the goal of inferring the authorship of these unsigned opinions. For each opinion, we trained a MaxEnt model using data from the nine sitting justices at the time; for example, in 2006, the training set consists of opinions from Justices Stevens, Scalia, Kennedy, Souter, Thomas, Ginsburg, Breyer, and Alito, along with Chief Justice Roberts. It is worth noting that the model's output probabilities for the most probable justice is often fairly low; for example, if we supplied a cutoff threshold of 0.43 to have 90% confidence in our prediction (as per Figure 2), the model would choose to "abstain" on predicting a justice in many of these cases.

Table 2.9: Predicted authorship of Roberts Court per curiam decisions

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
Continued on next page				

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
10/05/05	Dye v. Hofbauer	Kennedy (0.466)	Scalia (0.165)	Ginsburg (0.093)
10/17/05	Schiro v. Smith	O'Connor (0.207)	Thomas (0.192)	Scalia (0.152)
10/31/05	Eberhart v. United States	Thomas (0.268)	Scalia (0.189)	Ginsburg (0.165)
10/31/05	Kane v. Garcia Espitia	Scalia (0.190)	Thomas (0.174)	O'Connor (0.145)
11/28/05	Bradshaw v. Richey	O'Connor (0.325)	Scalia (0.294)	Thomas (0.115)
01/23/06	Wisconsin Right to Life, Inc. v. Federal Election Commission	Roberts (0.210)	O'Connor (0.151)	Stevens (0.129)
02/21/06	Ash v. Tyson Foods, Inc.	Kennedy (0.326)	Scalia (0.304)	Thomas (0.149)
02/21/06	Lance v. Dennis	Scalia (0.351)	Ginsburg (0.146)	Thomas (0.122)
02/21/06	Ministry of Defense and Support v. Elahi	Breyer (0.397)	Scalia (0.209)	Thomas (0.179)
04/17/06	Gonzales v. Thomas	Breyer (0.715)	Thomas (0.081)	Scalia (0.076)
04/24/06	Salinas v. United States	Roberts (0.171)	Breyer (0.152)	Scalia (0.151)

Continued on next page

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
06/05/06	Whitman v. Department of Transportation	Kennedy (0.248)	Scalia (0.180)	Souter (0.135)
06/19/06	Youngblood v. West Virginia	Scalia (0.193)	Ginsburg (0.182)	Thomas (0.126)
10/20/06	Purcell v. Gonzalez	Kennedy (0.574)	Ginsburg (0.142)	Stevens (0.058)
01/09/07	Burton v. Stewart	Thomas (0.452)	Roberts (0.229)	Scalia (0.182)
03/05/07	Lance v. Coffman	Scalia (0.306)	Roberts (0.300)	Alito (0.081)
05/21/07	Los Angeles County v. Rettele	Scalia (0.301)	Kennedy (0.189)	Stevens (0.111)
05/21/07	Roper v. Weaver	Thomas (0.330)	Kennedy (0.127)	Ginsburg (0.117)
06/04/07	Erickson v. Pardus	Ginsburg (0.335)	Scalia (0.214)	Thomas (0.146)
11/05/07	Allen v. Siebert	Thomas (0.292)	Scalia (0.261)	Roberts (0.153)
01/07/08	Arave v. Hoffman	Ginsburg (0.212)	Thomas (0.154)	Kennedy (0.152)
01/07/08	Wright v. Van Patten	Thomas (0.517)	Scalia (0.135)	Ginsburg (0.068)

Continued on next page

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
08/05/08	Medellin v. Texas	Kennedy (0.186)	Breyer (0.164)	Scalia (0.140)
10/14/08	Moore v. United States	Thomas (0.203)	Scalia (0.167)	Breyer (0.132)
12/02/08	Brunner v. Ohio Republican Party	Ginsburg (0.153)	Kennedy (0.139)	Scalia (0.126)
12/02/08	Hedgpeth v. Pulido	Thomas (0.256)	Roberts (0.211)	Breyer (0.140)
01/21/09	Spears v. United States	Scalia (0.531)	Roberts (0.201)	Thomas (0.121)
01/26/09	Nelson v. United States	Thomas (0.209)	Scalia (0.193)	Roberts (0.138)
06/01/09	CSX Transportation, Inc. v. Hensley	Roberts (0.179)	Kennedy (0.157)	Scalia (0.143)
06/09/09	Indiana State Police Pension Trust v. Chrysler LLC	Roberts (0.186)	Ginsburg (0.163)	Alito (0.121)
10/20/09	Concoran v. Levenhagen	Scalia (0.210)	Breyer (0.148)	Kennedy (0.140)
11/09/09	Bobby v. Van Hook	Breyer (0.206)	Kennedy (0.203)	Ginsburg (0.197)
11/16/09	Wong v. Belmontes	Kennedy (0.428)	Scalia (0.229)	Roberts (0.137)

Continued on next page

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
11/30/09	Porter v. McCollum	Thomas (0.229)	Scalia (0.191)	Kennedy (0.169)
12/07/09	Michigan v. Fisher	Roberts (0.244)	Scalia (0.178)	Alito (0.122)
01/11/10	McDaniel v. Brown	Thomas (0.768)	Kennedy (0.074)	Scalia (0.038)
01/13/10	Hollingsworth v. Perry	Kennedy (0.787)	Scalia (0.066)	Stevens (0.040)
01/19/10	Presley v. Georgia	Kennedy (0.619)	Stevens (0.089)	Scalia (0.068)
01/19/10	Wellons v. Hall	Scalia (0.440)	Thomas (0.276)	Ginsburg (0.064)
02/22/10	Thaler v. Haynes	Alito (0.248)	Thomas (0.238)	Ginsburg (0.179)
02/22/10	Wilkins v. Gaddy	Scalia (0.243)	Ginsburg (0.233)	Thomas (0.212)
03/01/10	Kiyemba v. Obama	Roberts (0.181)	Scalia (0.161)	Ginsburg (0.154)
05/24/10	Jefferson v. Upton	Scalia (0.330)	Breyer (0.239)	Thomas (0.231)
06/07/10	United States v. Juvenile Male	Thomas (0.249)	Roberts (0.194)	Scalia (0.141)

Continued on next page

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
06/29/10	Sears v. Upton	Thomas (0.223)	Breyer (0.159)	Scalia (0.157)
11/08/10	Wilson v. Corcoran	Scalia (0.324)	Thomas (0.201)	Ginsburg (0.186)
01/10/11	Madison County v. Oneida Indian Nation	Thomas (0.190)	Kennedy (0.139)	Ginsburg (0.139)
01/24/11	Swarthout v. Cooke	Scalia (0.254)	Roberts (0.190)	Thomas (0.184)
03/21/11	Felkner v. Jackson	Thomas (0.359)	Roberts (0.211)	Kennedy (0.177)
05/02/11	Bobby v. Mitts	Thomas (0.528)	Scalia (0.124)	Roberts (0.123)
07/07/11	Leal Garcia v. Tecas	Scalia (0.240)	Roberts (0.234)	Breyer (0.138)
10/31/11	Cavazos v. Smith	Kennedy (0.334)	Ginsburg (0.195)	Scalia (0.183)
11/07/11	Bobby v. Dixon	Scalia (0.373)	Kennedy (0.213)	Ginsburg (0.139)
11/07/11	KPMG LLP v. Cocchi	Thomas (0.285)	Kennedy (0.254)	Scalia (0.111)
12/12/11	Hardy v. Cross	Thomas (0.247)	Alito (0.223)	Ginsburg (0.186)

Continued on next page

Table 2.9 – continued from previous page

Date of Decision	Case	Highest probability justice	Second-highest probability justice	Third-highest probability justice
01/20/12	Perry v. Perez	Kennedy (0.344)	Roberts (0.230)	Scalia (0.184)
01/23/12	Ryburn v. Huff	Alito (0.212)	Ginsburg (0.185)	Scalia (0.176)
02/21/12	Wetzel v. Lambert	Roberts (0.239)	Thomas (0.190)	Kennedy (0.177)
05/29/12	Coleman v. Johnson	Thomas (0.225)	Kennedy (0.203)	Breyer (0.164)
05/29/12	Marmet Health Care Center, Inc. v. Brown	Thomas (0.324)	Kennedy (0.195)	Alito (0.139)
05/29/12	Parker v. Matthews	Kennedy (0.265)	Ginsburg (0.181)	Scalia (0.134)
06/25/12	American Tradition Partnership, Inc. v. Bullock	Thomas (0.232)	Roberts (0.161)	Kennedy (0.147)
09/25/12	Tennant v. Jefferson County	Roberts (0.603)	Kennedy (0.119)	Ginsburg (0.076)
11/05/12	Lefemine v. Wideman	Ginsburg (0.267)	Roberts (0.188)	Thomas (0.182)
11/26/12	Nitro-Lift Technologies, LLC v. Howard	Thomas (0.472)	Scalia (0.248)	Kennedy (0.080)

Assuming these predictions are accurate, they are provocative. Justices commonly described as "conservative" are predicted authors of 45 out of the 65 per curiam opinions (69.2%). Justices commonly described as "conservative-swing" are predicted authors of 13 of the remaining 20 opinions — 11 for Justice Kennedy and 2 for Justice O' Connor. Thus, conservative or conservative-swing justices are predicted authors of 58 out of the 65 per curiam opinions (89.2%). Justices commonly described as liberal are predicted authors of only 7 of the opinions (10.8%). Table 2.10 provides a yearly break down of the predicted authors by their ideology.

Table 2.10: Predicted author ideology of per curiam opinions by year

Year	Total	Conservative	Conservative-swing	Liberal
2012	2	1 (50%)	0	1 (50%)
2011	12	9 (75%)	3 (25%)	0
2010	14	10 (71.4%)	3 (21.4%)	1 (7.1%)
2009	14	10 (71.4%)	3 (21.4%)	1 (7.1%)
2008	7	6 (85.7%)	0	1 (14.3%)
2007	4	2 (50%)	1 (25%)	1 (25%)
2006	5	3 (60%)	1 (20%)	1 (20%)
2005	14	7 (50%)	5 (35.7%)	2 (14.3%)

Excluding 2012, in which there were only two per curiam opinions, conservative and conservative-swing justices in the Roberts Court are predicted authors of between 75% and 100% of the per curiam decisions per year.

2.7 Conclusion

Machine learning techniques can be used to attribute authorship of judicial opinions. We have demonstrated that word-level features can distinguish authorship with substantial accuracy. The inferred authorship for the opinions in Sebelius provides unambiguous quantitative support for one theory of authorship offered in the media. Applying these methods to the unsigned per curiam opinions of the Roberts Court yields provocative results. The performance of the model on test opinions, along with the stylistic features that it uses to determine performance, suggests that it could be useful for other courts. Overall, our work underscores the broad applicability of nat-

ural language processing tools to yield quantitative insights into issues traditionally studied only qualitatively and manually.

Chapter 3

Law Is Code: A Software Engineering Approach to Analyzing the United States Code

Preamble

This chapter is largely adapted from the following paper:

Li, W., Azar, P., Larochelle, D., Hill, P., Lo, A.W. Law Is Code: A Software Engineering Approach to Analyzing the United States Code. Journal of Business and Technology Law, 10, 297, 2015 [29].

3.1 Context

This chapter introduces a *vocabulary* for analyzing legal code like software code: Inspired by the common usage of the word "code" in the domains of law and computer science, we present a set of metrics and techniques inspired from the study of software engineering quality that attempt to measure the quality and complexity of laws. These metrics include simply counting words, network analyses of the cross-references in the United States Code, and measurements of the number the conditional statements in the law. For the overall goal of this thesis, the most relevant part of this chapter is

the work in measuring how much of the text of the U.S. Code persists or changes time, as measured by the amount of text reuse across versions. The techniques used to measure text reuse have significant conceptual and technical overlap with the work in the next two chapters.

3.2 Abstract

The agglomeration of rules and regulations over time has produced a body of legal code that no single individual can fully comprehend. This complexity produces inefficiencies, makes the processes of understanding and changing the law difficult, and frustrates the fundamental principle that the law should provide fair notice to the governed. In this article, we take a quantitative, unbiased, and software-engineering approach to analyze the evolution of the United States Code from 1926 to today. Software engineers frequently face the challenge of understanding and managing large, structured collections of instructions, directives, and conditional statements, and we adapt and apply their techniques to the U.S. Code over time. Our work produces insights into the structure of the U.S. Code as a whole, its strengths and vulnerabilities, and new ways of thinking about individual laws. For example, we identify the first appearance and spread of important terms in the U.S. Code like "whistleblower" and "privacy." We also analyze and visualize the network structure of certain substantial reforms, including the Patient Protection and Affordable Care Act ("PPACA") and the Dodd-Frank Wall Street Reform and Consumer Protection Act, and show how the interconnections of references can increase complexity and create the potential for unintended consequences. Our work is a timely illustration of computational approaches to law as the legal profession embraces technology for scholarship, to increase efficiency, and to improve access to justice.

3.3 Introduction

Laws and regulations are the rules by which societies operate. Beginning with the Code of Ur-Nammu more than 4,000 years ago,[1] societies have often formalized laws and regulations by recording them in written form.[2] Over time, this simple custom evolved, producing some of the most significant innovations in the history of civilization, including replacing the rule of monarchs with the rule of law.[3]

With the rule of law flourishing in modern societies, subtler challenges have emerged as the unintended consequences of an unwieldy system of laws. The agglomeration of rules and regulations over time and across the many facets of social, political, and economic interactions has produced a body of legal code that no single individual can fully comprehend. Despite the fact that laws now apply to virtually every aspect of daily life, the sheer volume of code requires citizens to have a certain degree of faith in the experts with whom we have entrusted the responsibilities of creating, managing, analyzing, and ultimately applying that code.

The increasing complexity of the legal system has several important implications. First, it produces inefficiencies.[4] The time, money, and other human resources associated with the rule of law in modern society are substantial and growing.[5] Second, as the legal code expands in size, interactions between provisions will quickly outstrip humans' ability to manage them using traditional methods.[6] Third, if one purpose of a legal code is to provide notice to the governed, then that purpose is frustrated

[1] See, e.g., J.J. Finkelstein, *The Laws of Ur-Nammu* , 22 J. of Cuneiform Stud. 66 (1968-69), *available at* `http://www.jstor.org/discover/10.2307/1359121?uid=3739696&uid=2&uid=4&uid=3739256&sid=21104628208137` (reprinting Ur-Nammu's legal code).

[2] See LEGAL TRADITIONS OF THE WORLD: SUSTAINABLE DIVERSITY IN LAW 100-01, 135 (4th ed. 2010).

[3] *See*, e.g., Russell Fowler, *The 800th Anniversary of Magna Carta: A Time for Lawyers to Remember* , 50 TENN. B.J. 23 (2014) (commemorating the Magna Carta, which limited the British monarch's power).

[4] Susan Hayes Stephan, *Blowing the Whistle on Justice as Sport: 100 Years of Playing a Non-Zero Sum Game* , 30 HAMLINE L. REV. 587, 588 (2007) (analogizing litigation to an inefficient "game").

[5] See LAWYERS FOR CIVIL JUSTICE ET AL., LITIGATION COST SURVEY OF MAJOR COMPANIES 2-6 (2010), *available at* `http://perma.cc/4NCU-W766`

[6] *See* Dru Stevenson, Costs of Codification, 2014 U. ILL. L. REV. 1129, 1129 (2014) (identifying downsides to codification, including "legislative borrowing," overcriminalization, unmanageable legal-information costs, and judicial overemphasis on statutory text rather than policy).

when the code becomes increasingly opaque to the vast majority of citizens.[7]

In this Article, we propose new quantitative methods for understanding and managing the system that comprises the entire legal code. We start from the well-trodden premise that legal code is in many respects similar to computer source code.[8] In this Article, we conclude that "law is code." While Lessig likens software code to the laws of nature, this Article analogizes source code to legal code and then proposes legal reforms. We take a computational approach to studying the full text of the United States Code from its first edition in 1926 to the present day. Our approach adopts techniques that software engineers use to analyze the evolution and structure of large software codebases, which are often millions of lines in length.[9] In particular, we examine the rise and fall in usage of specific words and phrases in the U.S. Code, quantify the amount of change over time, and present metrics and visualizations of its cross-reference structure. Our work leads to novel and provocative analyses of the U.S. Code's systemic structure, insights into its strengths and weaknesses, and new ways of thinking about the nature of individual laws. For example, we identify the first appearance and spread of important terms like "whistleblower" and "privacy."[10] Also, we visually represent laws' network structures, to show how certain laws that introduce substantial reform, including the Patient Protection and Affordable Care Act ("PPACA") and the Dodd-Frank Wall Street Reform and Consumer Protection Act,[11] differ from other long pieces of legislation, such as appropriations bills.[12]

We structure this chapter as follows: Section 3.4 summarizes past and current

[7] *See* BRADFORD J. WHITE & PAUL W. EDMONDSON, PROCEDURAL DUE PROCESS IN PLAIN ENGLISH: A GUIDE FOR PRESERVATION COMMISSIONS (3d ed. 2008) (lamenting that lay persons must rely on lawyers to understand the law); *see also* Mathews v. Eldridge, 424 U.S. 319, 348-49 (1976) (delineating procedural due process requirements, including notice).

[8] *See* LAWRENCE LESSIG, CODE AND OTHER LAWS OF CYBERSPACE 6 (1999). Lines of software code are to cyberspace what the laws of physics are to the non-virtual world; they determine what is possible and, in turn, what can be regulated. Dan Orr, *Book Review: Code and Other Laws of Cyberspace*, RES. CENTER FOR CYBERCULTURE STUD. (Aug. 2000), `http://rccs.usfca.edu/bookinfo.asp?BookID=79&ReviewID=79`

[9] *See infra* Sections 3.6 and 3.7

[10] *See infra* Section 3.6

[11] Patient Protection and Affordable Care Act, Pub. L. No. 111-148, 124 Stat. 119 (2010); Dodd-Frank Wall Street Reform and Consumer Protection Act, Pub. L. No. 111-203, 124 Stat. 1376 (2010).

[12] *See infra* Section 3.8

codification efforts in the United States.[13] Section 3.5 describes our U.S. Code dataset, provides an overview of key software engineering principles that we adopt, and outlines the analytics and algorithms that we use.[14] Part IV applies these tools to explore the U.S. Code's *evolution* ; for example, we quantify the rise and fall of key terms and the percentage change in selected titles' content.[15] In Part V, we focus on recently passed laws' impact on the U.S. Code.[16] Finally, Part VI applies software engineering metrics to specific titles of the current U.S. Code, focusing on Title 12 (Banks and Banking) and Title 26 (Internal Revenue Service) as examples.[17] In the appendices, we define our network-analysis metrics and visually represent different kinds of laws' network structures.

3.4 The United States Code

Given the frequency with which legal practitioners and scholars cite the U.S. Code, many facts about the U.S. Code may seem shocking. For example, Congress did not authorize the official collection of federal statutes until 1926,[18] meaning that as of 2012, around 5 million living U.S. citizens were born before the U.S. Code was first published.[19] Until 1947, the U.S. Code was merely *prima facie* evidence of the statutes reproduced within the U.S. Code;[20] only after 1947 did Congress begin the slow, piecemeal process of converting the U.S. Code into controlling law (known as "positive law codification").[21] Further, the U.S. Code is *still* only *prima facie* evidence

[13] *See infra* Section 3.4.
[14] *See infra* Section 3.5.
[15] *See infra* Part IV.
[16] *See infra* Part V.
[17] See infra Part VI.
[18] CODE OF LAWS OF THE UNITED STATES OF AMERICA OF A GENERAL AND PERMANENT NATURE IN FORCE DECEMBER 7, 1925; see Will Tress, *Lost Laws: What We Can't Find in the United States Code* , 40 GOLDEN GATE U. L. REV. 129, 136 (2010).
[19] See U.S. CENSUS BUREAU, CURRENT POPULATION SURVEY, ANNUAL SOCIAL AND ECONOMIC SUPPLEMENT, 2012 (2013), available at `http://www.census.gov/population/age/data/files/2012/2012gender_table1.xlsx.`
[20] See Tress, supra note 18, at 137-38.
[21] Act of July 30, 1947, ch. 388, 61 Stat. 633, 638; *Positive Law Codification* , OFFICE OF THE LAW REVISION COUNSEL: U.S. CODE, `http://uscode.house.gov/codification/legislation.shtml` (last visited Jan. 27, 2015); *see* Tress, *supra* note 18, at 137

of the law for 26 of the U.S. Code's 52 titles.[22]

The goal of the U.S. Code is simple enough: to provide "the laws of the United States, general and permanent in their nature."[23] The project of codification, however, has been wrought with difficulty from the beginning.[24] This Part outlines the goals that lawmakers have aspired to address with codification and the troubles they have encountered along the way. The techniques proposed in this Article use modern computer scientific methods to analyze and remedy issues that have plagued U.S. lawmakers for centuries.

3.4.1 Early Federal Codification Problems

In 1795, Congress authorized the first compilation of federal statutes, which included all public laws and treaties to date.[25] But until 1845, the annual session laws were not published on a regular basis; rather, federal statutes were published in newspapers.[26] In the early 1820s, individual states began to debate the idea of codification, with the New York Revised Code of 1829 leading the way, followed by newly admitted western states.[27] Recognizing the value of codification, private publishers produced chronological, bound volumes of U.S. public laws.[28]

The first federal solution came when Little, Brown & Co., a Boston-based private publisher, proposed the creation of the Statutes at Large in 1845.[29] This collection of laws, as updated, remains the authoritative collection for half the U.S. Code titles today.[30] The Statutes at Large contain a chronological set of laws which Congress

[22] See *Positive Law Codification* , *supra* note 21.

[23] 1 U.S.C. §204(a) (2012).

[24] *See* Tress, *supra* note 18, at 133-38.

[25] Act of Mar. 3, 1795, ch. 50, 1 Stat. 443.

[26] *See* Ralph H. Dwan & Ernest R. Feidler, *The Federal Statutes-Their History & Use* , 22 MINN. L. REV. 1008, 1010 (1938); Tress, *supra* note 18, at 133.

[27] See CHARLES COOK, AMERICAN CODIFICATION MOVEMENT 158-59 (1981); EDWIN C. SURRENCY, A HISTORY OF AMERICAN LAW PUBLISHING 90-91 (1990).

[28] Richard J. McKinney, *Basic Overview on How Federal Laws Are Published, Organized and Cited* , LAW LIBRARIANS' SOC'Y WASH. D.C. 2 (Jan. 12, 2006), `http://www.llsdc.org/assets/sourcebook/federal-laws.pdf` .

[29] *Id.*

[30] See *Positive Law Codification* , *supra* note 21.

passed and the President signed into law.[31] Each volume of the Statutes at Large covered one congressional session.[32] The Government Printing Office—created in 1861—replaced Little, Brown & Co. as the entity responsible for publishing the Statutes at Large until 1950, when the Office of the Federal Register in the National Archives took over.[33]

While the Statutes at Large improved matters by providing a definitive collection of laws, the chronological, session-based presentation, among other sundry conventions, made it difficult for lawmakers to determine what was current U.S. law in any given subject area.[34] In 1848, the chairman of the House Judiciary Committee proposed a bill to revise the session laws.[35] The accompanying House Report outlined a litany of issues, including that the session laws may have been "enacted under the pressure of momentary emergency; if not inconsistent, they are obscure; sometimes involved in statutes dissimilar in title and object, and always scattered over different parts of a broad surface, in the numerous hiding places of which they are concealed."[36] The report admonished, with some prescience, that "enactments defining the duties of a particular office should naturally be so united as to furnish all needful information in one comprehensive body. That which seems to be complete in its enumeration should be so in reality."[37]

In 1866, Congress created a commission tasked "to revise, simplify, arrange, and consolidate all statutes of the United States, general and permanent in their nature."[38] Two years into their task, the commission reported several insurmountable difficulties, noting "[w] here several statutes relating to the same subject modify each other, it has been impossible to state their united effect without writing a new statute."[39] In 1872,

[31] *See* McKinney, *supra* note 28, at 2.
[32] *Id.* at 3.
[33] *Id.*
[34] *Id.*
[35] *See* H.R. 535, 30th Cong. (1st Sess. 1848). No record of the bill remains. Tress, *supra* note 18, at 133.
[36] H.R. REP. NO. 30-671, at 1 (1848).
[37] *Id.* at 2.
[38] Act of June 27, 1866, ch. 140, 14 Stat. 74.
[39] WILLIAM JOHNSTON & CHARLES P. JAMES, REPORT OF THE COMMISSIONERS APPOINTED UNDER ACT OF JUNE 27, 1866, S. Misc. Doc. 101, 40th Cong. (2d Sess. 1868).

the commission presented its proposed revisions, which Congress deemed too extreme a departure from the language of existing laws, and delegated the draft to a special reviser charged with reversing much of the commission's proposals.[40] Ultimately, this process yielded the Revised Statutes of the United States, containing 70 titles, which revised, reorganized, and consolidated all permanent and general U.S. laws, and was enacted in 1874 and published in 1875.[41]

The Revised Statutes repealed all general acts "embraced in any section" of the revisions, replacing them as controlling authority.[42] Shortly after publication, however, numerous mistakes and omissions were identified.[43] Congress addressed these errors in an amended and updated 1878 revision.[44] Sensitive to the debacle that these errors and omissions produced, the 1878 Revision provided that it would not "preclude reference to, nor control, in case of discrepancy, the effect of any original act passed by Congress since" December 1, 1873.[45]

Problems arising from the Revised Statutes dealt a blow to the codification movement. The ensuing 50 years saw several proposals to update or replace the Revised Statutes, but Congress did not issue another code until 1926.[46] In the interim, private publishers again shouldered the collection and organization of laws passed since the 1878 Revisions.[47]

3.4.2 Early Problems with the U.S. Code

Perhaps still stinging from prior codification efforts, Congress undertook several measures to forestall similar issues. First, Congress enlisted the professional expertise of two private code publishers, West and Edward Thompson, to oversee the new edition

[40]Dwan & Feidler, *supra* note 26, at 1013.

[41]REVISED STATUTES OF THE UNITED STATES PASSED AT THE FIRST SESSION OF THE FORTY-THIRD CONGRESS 1873-74 (2d ed. 1878); *see* McKinney, *supra* note 28, at 3.

[42]Sec. 559, 1 Rev. Stat. 1091 (1873); *see* Tress, *supra* note 18, at 135.

[43]INACCURACIES AND OMISSIONS IN REVISED STATUTES, H. Exec. Doc. 36, 44th Cong. (1st Sess. 1876).

[44]Revised Statutes (1878).

[45]*Id.* at iii (preface).

[46]CODE OF LAWS OF THE UNITED STATES OF AMERICA OF A GENERAL AND PERMANENT NATURE IN FORCE DECEMBER 7, 1925.

[47]SURRENCY, supra note 27, at 107-10; Dwan & Feidler, supra note 26, at 1016-21.

of the Code.[48] Second, Congress was careful to note that the 1926 Revisions were an "official restatement in convenient form" of U.S. law, but "[n] o new law is enacted and no law repealed. It is *prima facie* the law. It is presumed to be the law. The presumption is rebuttable by production of prior unrepealed Acts of Congress at variance with the Code."[49]

Third, this overly cautious, mostly redundant preface was the product of legislative compromise. The original bill, as passed by the House,[50] provided that the U.S. Code would remain *prima facie* evidence until June 30, 1927, at which time it would become controlling law.[51] Lawmakers hoped that window would allow sufficient time to correct any new errors.[52] Fearing the prospect of errors, however, the Senate amended the bill to prevent the U.S. Code from becoming the controlling statement of the law.[53] True enough, 537 errors were later found and corrected, 88 of which were substantive errors.[54]

Identifying those errors also presented difficulty. The aforementioned preface cautiously limited the U.S. Code to *prima facie* evidence of U.S. law, but it failed to identify which published laws *could be* cited to rebut the presumption.[55] Ultimately, the 1878 Revision controlled for statutes enacted before December 1, 1873, and although the 1878 Revision also contained statements of the law from 1874 to 1878, the Statutes at Large were the authoritative text for all statutes from 1873 to date.[56]

---

[48] Tress, *supra* note 18, at 136.

[49] Preface, U.S.C. (1926).

[50] H.R. 10000, 69th Cong. (1926).

[51] See Richard J. McKinney, *Unraveling the Mysteries of the U.S. Code* , LAW LIBRARIANS' SOC'Y WASH. D.C. 1 (Aug. 2009), `http://www.llsdc.org/assets/sourcebook/usc-mysteries.pdf` .

[52] *Id.*

[53] *See* Act of June 30, 1926, ch. 712, 44 Stat. 777.

[54] *See* McKinney, *supra* note 51, at 1.

[55] Tress, *supra* note 18, at 137. Private publishers like West and Lexis filled the gap by providing annual updates, and today Congress annually archives electronic versions. *Id.* at 137 n.42.

[56] *Id.* at 137.

3.4.3 The U.S. Code, 1926 to Today

After its first publication in 1926, the U.S. Code was replaced by a new edition in 1934, followed by new editions every six years.[57] The U.S. Code remained only *prima facie* evidence of the law until 1947, when Congress began the process of converting the U.S. Code to the controlling statement of the law.[58] That year, U.S. Code Title 1 (General Provisions) was positively codified, along with Title 4 (Flag & Seal, Seat of Government, and the States), Title 6 (Official & Penal Bonds), Title 9 (Arbitration), and Title 17 (Copyrights).[59] Interestingly, one congressman noted the intent to begin with "the more important titles and those urgently needing codification," including, for example, Title 28 on the Judiciary.[60] Despite that lofty initial goal, the first few positively codified titles were "low-hanging fruit" that required little editing to prepare[61] —a volte-face that was likely motivated by Congress's prior track record with positive law codification.[62]

The original version of the U.S. Code organized then-existing federal laws into 50 titles within a single bound volume; today, the U.S. Code contains over 47,000 pages, 51 titles, and spans several volumes.[63] In 1974, Congress created the Office of the Law Revision Counsel of the U.S. House of Representatives ("OLRC") to prepare and publish the U.S. Code.[64] Among other things, the OLRC (1) periodically reviews enacted laws and makes recommendations for repealing obsolete, superfluous, and superseded provisions; (2) determines whether and how new laws should be incorporated into the code; (3) classifies the newly enacted provisions so that they may

[57] *Id.* ; see 1 U.S.C. §202(c) (2012).

[58] Tress, *supra* note 18, at 137; *see* Act of July 30, 1947, ch. 388, 61 Stat. 633, 638.

[59] Tress, *supra* note 18, at 137-38 (citing William Chamberlain, *Enactment of Parts of the United States Code into Positive Law* , 36 GEO. L.J. 217 (1947)).

[60] 93 CONG. REC. 8384 (1947) (remarks of Rep. John M. Robison).

[61] Tress, *supra* note 18, at 138.

[62] 2 U.S.C. §285b (2012); *About the Office; Contact Information* , OFFICE OF THE LAW REVISION COUNSEL: U.S. CODE, `http://uscode.house.gov/about_office.xhtml` (last visited Jan. 22, 2015).

[63] Peter LeFevre, *Positive Law Codification Will Modernize U.S. Code* , THE HILL (Sept. 28, 2010, 5:33 PM), `http://thehill.com/blogs/congress-blog/judicial/121375-positive-law-codification-will-modernise-us-code.`

[64] Act of Dec. 27, 1974, ch. 3, Pub. L. No. 93-554, 88 Stat. 1771, 1777 (codified as amended at 2 U.S.C. §285 (2012)).

be incorporated into the relevant titles of the U.S. Code; (4) makes the necessary revisions to each title within the U.S. Code; and (5) recommends certain titles for positive law codification.

While the OLRC's task of incorporating new law into the U.S. Code can be simple when the laws are small and narrow in subject-matter (though not necessarily so), the task is more complicated when the laws are large, cover a multitude of subjects, and/or contain a complicated mixture of amendatory and freestanding provisions, general specific provisions, and permanent and temporary provisions.[65] The OLRC often must make impressionistic determinations about how to incorporate a new piece of legislation into the U.S. Code.[66] Moreover, while the OLRC can editorially add new sections, chapters, and statutory notes to 23 *non* -positive law titles, only Congress can add new sections and chapters to the 27 positive law titles, and only by amendment.[67]

3.4.4 Criticisms and Aspirations for the U.S. Code

Commentators cite numerous problems with the U.S. Code, including: (1) many of the new laws passed since 1926 are often shoehorned awkwardly into pre-existing titles; (2) Congress often pays little or no attention to existing laws when enacting new legislation, which makes it difficult for U.S. Code editors to keep statutes that relate to similar subjects together; and (3) the increasingly voluminous body of legislation since 1926 has produced many obscure, obsolete, and redundant provisions, archaic and inconsistent language, and statutory errors.[68]

In touting the benefits of positive law codification, the OLRC has identified and, in some cases, reaffirmed the U.S. Code's deficiencies. For example:

[65] *About Classification of Laws to the United States Code* , OFFICE OF THE LAW REVISION COUNSEL: U.S. CODE, `http://uscode.house.gov/about_classification.xhtml` (last visited Feb. 2, 2015).

[66] *Id.*

[67] *Id.*

[68] See LeFevre, *supra* note 62; *see also* Positive Law Codification, *supra* note 21 (noting that revisers seek to reorganize existing provisions, conform style and terminology, modernize obsolete language, and correct drafting errors); *About Classification of Laws to the United States Code* , *supra* note 65.

Improved organization. Provisions that are closely related by subject may be scattered in different places in the Code. Such provisions may have been enacted many years apart and incorporated at different times. Positive law codification affords an opportunity to revisit the organizational structure of statutory material. Thoughtful regrouping of provisions often yields a statutory product that is easier to use and that fosters a more comprehensive understanding of the law.

Elimination of obsolete provisions . Obsolete provisions are frequently identified in the course of preparing a positive law codification bill [and] are eliminated from the law after appropriate vetting of proposed changes. Although such changes seem small and innocuous when viewed individually, the cumulative effect of removing all obsolete provisions can be profound, resulting in a much more compact and comprehensible text.

Precise statutory text. The process of positive law codification promotes public access to the precise text of Federal statutory law. Provisions set out in non-positive law titles . . . may vary slightly from the precise language enacted into law; cross references are adapted and stylistic changes are made in order to facilitate the integration of Federal Statutory provisions.

Cleaner amendments. Positive law codification promotes accuracy and efficiency in the preparation of amendment specifying words to be struck or the place where new words are to be inserted or simplified; understanding the impact of proposed amendments is easier; drafting errors are reduced. In addition, compliance with congressional rules requiring comparative prints (showing proposed omissions and insertions) is facilitated.[69]

With the foregoing in mind, this Article's remaining parts describe the development and application of computer scientific techniques to assess and remedy problems that have plagued, and often continue to plague, the U.S. Code.

[69]See *Positive Law Codification in the United States Code* , OFFICE OF THE LAW REVISION COUNSEL: U.S. CODE 5, `http://uscode.house.gov/codification/positive_law_codification.pdf` (last visited Jan. 27, 2015).

3.5 Software Engineering Approaches to Analyzing the Law

3.5.1 Analogizing Legal Code to Software Code

Many analogies between software code and legal code apply at both general and specific levels. At a general level, both forms of code consist of a collection of rules that govern certain operations: human transactions in the case of legal code, and computer transactions in the case of computer code. The main difference—that humans interpret and implement laws whereas machines interpret and implement software—is more a matter of degree than kind. Because humans are more flexible and intelligent, laws need not be as explicit and precise as software. This lack of precision, however, is not without cost, as evidenced by the fundamental debate over "rules versus standards."[70] At a functional level, software and legal code share common features, functions, and frailties, irrespective of whether they are meant for or interpreted by humans or machines; hence, methods that have been developed in one domain should be relevant in the other. For example, based on concerns raised about the understandability of the law, we adopt four approaches from good software design practices—conciseness,[71] change,[72] coupling,[73] and complexity[74] —that should also have implications for good legal coding practices.

Software engineers apply a range of techniques to analyze computer code that may

[70] *See* Louis Kaplow, *Rules Versus Standards: An Economic Analysis* , 42 DUKE L.J. 557, 559-63 (1992).

[71] Harry H. Porter III, *Designing Programming Languages for Reliability 2* (Oct. 16, 2001) (unpublished paper) (on file with the Journal of Business & Technology Law), *available at* `http://web.cecs.pdx.edu/~harry/musings/RelLang.pdf` ("Most programming languages tend to emphasize conciseness.").

[72] *The Importance of Writing Good Code* , GNOME DEVELOPER, `https://developer.gnome.org/programming-guidelines/stable/writing-good-code.html.en` (last visited Jan. 21, 2015) ("General-purpose code is easier to reuse and modify than very specific code with lots of hardcoded assumptions.").

[73] Kailash Patidar et al., *Coupling and Cohesion Measures in Object Oriented Programming* , 3 INT'L J. ADVANCED RES. IN COMPUTER SCI. & SOFTWARE ENG'G 517, 517 (2013) ("[C]oupling is an important aspect in the evaluation of reusability and maintainability of components or services.").

[74] Neil D. Jones, COMPUTABILITY & COMPLEXITY: FROM A PROGRAMMING PERSPECTIVE, *at* vii (1997).

be relevant to analyzing the U.S. Code. First, conceptually, the similarity between software code and the U.S. Code in terms of function is the first important parallel that exists between computer programming and lawmaking. For example, software code often is written to compute some kind of output upon receiving certain inputs, *e.g.* , a computation module receives numerical values to perform arithmetic, and a search engine—like Google—receives a search query and returns a list of results. Similarly, the U.S. Code is a collection of laws that describes the inputs that determine when the authority of the federal government is to be applied and the outcomes that result; *e.g.* , how the salaries of members of Congress are determined (contained in Title 2) and the role of the U.S. Patent and Trademark Office (contained in Title 35).[75]

Second, the internal structure and composition of both software code and legal code also matter. Laws should be easy to read and comprehend so that individual citizens can understand their rights and obligations, and lawyers, legislators, judges, and jurors can more efficiently perform their jobs. Consonant with this concern, the U.S. Senate's Legislative Drafting Manual emphasizes readability.[76] Section 107, entitled "Focus on Reader," states: "A draft must be understandable to the reader. The rules in this manual should be applied in a manner that makes the draft clearer and easier to understand."[77] Similarly, the manual for the U.S. House of Representatives states, "Draft should be clear and understandable – In almost all cases, the message has a better chance of accomplishing your client's goal if it is readable and understandable. It should be written in English for real people."[78] Unnecessarily complicated laws can interfere with commerce, economic growth, and access to justice.[79]

[75] 2 U.S.C. §31 (2012); 35 U.S.C. §2 (2012) (delineating the U.S. Patent and Trademark Office's powers and duties).

[76] OFFICE OF THE LEGISLATIVE COUNSEL, U.S. SENATE, LEGISLATIVE DRAFTING MANUAL (1997), available at `http://www.law.yale.edu/documents/pdf/Faculty/SenateOfficeoftheLegislativeCounsel_LegislativeDraftingManual(1997).pdf` [hereinafter SENATE DRAFTING MANUAL].

[77] *Id.* at 7.

[78] THE OFFICE OF THE LEGISLATIVE COUNSEL, U.S. HOUSE OF REPRESENTATIVES, HOUSE LEGISLATIVE COUNSEL'S MANUAL ON DRAFTING STYLE 5 (1995), *available at* `http://legcounsel.house.gov/pdf/draftstyle.pdf` [hereinafter House Drafting Manual]

[79] *See Over-Regulated America* , ECONOMIST MAG., Feb. 18, 2012, *available at* `http://www.economist.com/node/21547789` (contending that the Dodd-Frank Act's complicated provi-

Because the form—not just the function—of the legal code is important, a software engineering approach can yield several new insights when applied to the law. In particular, software developers are deeply invested in making their code readable and easy to understand. Software engineering teams often need to integrate new team members, fix bugs, and refactor existing code, which are all tasks that require a deep understanding of code written by others who are often no longer available to provide support or clarification.[80] These requirements suggest that the tools used by software engineers to track progress, monitor potential vulnerabilities, or simply gain an understanding of an existing software codebase may be useful for serving the same functions when applied to legal code.

3.5.2 U.S. Code Datasets for Analysis

We use two datasets for our analyses:

1. We obtained complete text versions of the U.S. Code from 1926 to 2006 under license from William S. Hein & Co.[81] This dataset includes the editions of the U.S. Code from 1926, 1934, 1940, 1946, 1952, 1958, 1964, 1970, 1976, 1982, 1988, 1994, 2000, and 2006. The text in these U.S. Code editions is split up only per title, meaning that most of our analyses and visualizations are done on a title level. Collectively, we refer to the U.S. Code editions from Hein as our "historical dataset."

2. The official current U.S. Code is available for free download from the Office of the Law Revision Counsel (OLRC).[82] This version is in Extensible Markup

sions constrict economic growth); Michael Burgess, *Death Is Much Less Complicated Than the U.S. Tax Code* , Forbes (Apr. 15, 2013, 8:00 AM), `http://www.forbes.com/sites/realspin/2013/04/15/death-is-much-less-complicated-than-the-u-s-tax-code/` (arguing that the federal tax code confuses taxpayers and unfairly advantages certain groups).

[80] Robert Sedgewick & Kevin Wayne, INTRODUCTION TO PROGRAMMING IN JAVA 8 (2008) (Bugs "are the bane of a programmer's existence: the error messages can be confusing or misleading, and the source of the error can be very hard to find").

[81] *See U.S. Code* , HEINONLINE, `http://heinonline.org/HOL/Index?collection=uscode` (last visited Feb. 2, 2015).

[82] *Current Release Point* , Office of the Law Revision Counsel: U.S. Code, `http://uscode.house.gov/download/releasepoints/us/pl/113/290not235not287/xml_uscAll@113-290not235not287.zip` (last visited Jan. 21, 2015).

Language (XML) format, which means that headings, sub-headings, and cross-references are annotated within the document.[83] As a result, in addition to analyzing the text of the U.S. Code on a per-title basis, we can also use algorithmic approaches on a per-section basis. We also used data from the Cornell Legal Information Institute (LII), which is similarly structured, for data mining the cross-references from the U.S. Code.[84] We refer to the current U.S. Code edition from the OLRC as our "current dataset."

3.5.3 Choosing Software Engineering Approaches and Metrics

The software engineering industry uses a wide range of frameworks, principles, and metrics in its work.[85] Some materials focus on "design patterns," which describe solutions to common programming tasks;[86] others emphasize project-management techniques to monitor a software engineering project's progress;[87] and still others educate coder-readers with examples of poorly written code.[88] There are entire treatises on subtypes of software engineering, such as refactoring, in which a codebase is re-organized so that it is cleaner and easier to understand.[89] For the purposes of this chapter, we focus on four categories of issues that affect the understandability of the legal code: conciseness, change, coupling, and complexity.

Conciseness. According to the U.S. Senate Legislative Drafting Manual, brevity is desirable: "Use short, simple sentences rather than complex or compound sentences.

[83] *United States Legislative Markup: User Guide for the USLM Schema* , OFFICE OF THE LAW REVISION COUNSEL: U.S. CODE, `http://uscode.house.gov/download/resources/USLM-User-Guide.pdf` (last visited Feb. 2, 2015) [hereinafter *USLM User Guide*] .

[84] *U.S. Code: Table of Contents* , CORNELL LEGAL INFO. INST., `http://www.law.cornell.edu/uscode/text` (last visited Feb. 22, 2015).

[85] *See* Steve McConnell, CODE COMPLETE: A PRACTICAL HANDBOOK FOR SOFTWARE CONSTRUCTION (2d ed. 2004).

[86] *See, e.g.* , Erich Gamma et al., DESIGN PATTERNS: ELEMENTS OF REUSABLE OBJECT-ORIENTED SOFTWARE 2–4 (1995) (defining a design pattern as a solution to a common problem that one can use "a million times over, without ever using it the same way twice").

[87] *See, e.g.* Barbara Kitchenham, SOFTWARE METRICS: MEASUREMENT FOR SOFTWARE PROCESS IMPROVEMENT 5 (1996).

[88] *See, e.g.* Robert C. Martin, CLEAN CODE: A HANDBOOK OF AGILE SOFTWARE CRAFTSMANSHIP 285–314 (2009) (listing common coding problems).

[89] *See, e.g.,* Martin Fowler, REFACTORING: IMPROVING THE DESIGN OF EXISTING CODE, at xvii (1999) (explaining how to refactor without "introduc[ing] bugs into the code").

If a shorter term is as good as a longer term, use the shorter term."[90] Laws that are long and verbose require more time to read, interpret, and revise. Despite being a simple and limited metric, length is a reasonable starting point for quantifying legal code.

In software engineering, the size of a software codebase, usually measured by lines of code ("LOC"), is a common metric for evaluating the effort required to develop and maintain it.[91] Each line of code has the potential to contain errors or unnecessary complexity. Large amounts of code, therefore, correspond to larger, more complicated software, which might have a greater number of bugs. In practice, while imperfect, counting lines of code is a simple, reasonable starting point to start characterizing a codebase's complexity and potential problems.[92]

In software code, the number of LOCs is typically used as the rough approximation of complexity; since most programming languages generally require line breaks, this provides a rough indication of the number of "instructions" in the program.[93] Turning to the U.S. Code, to measure conciseness, we use the number of words as our measurement unit because each clause or sentence is not necessarily a new line in the document.

We count words in two ways for different datasets. First, for our historical dataset, we visualize the length of different titles of the U.S. Code at different snapshots, namely every six years when a new complete edition of the U.S. Code is released. For the most current edition of the U.S. Code, we compare the lengths of different bills and titles.

Change. Revisions to the law require interested parties to understand what has changed and what has remained the same. Changes may also introduce unexpected or

[90] SENATE DRAFTING MANUAL, *supra* note 76, at 4.

[91] McConnell, *supra* note 85, at 725–26.

[92] *See* Jarrett Rosenberg, *Some Misconceptions About Lines of Code* , *in* PROCEEDINGS FOURTH INTERNATIONAL SOFTWARE METRICS SYMPOSIUM 137 (1997), *available at* `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=637174.` .

[93] Graylin Jay et al., *Cyclomatic Complexity and Lines of Code: Empirical Evidence of a Stable Linear Relationship* , 2 J. SOFTWARE ENG'G & APPLICATIONS 137, 137 (2009), *available at* `http://www.scirp.org/Journal/PaperDownload.aspx?paperID=779` (finding a "practically perfect linear relationship" between lines of code and cyclomatic complexity).

unintended effects. Over the span of many decades, the U.S. Code has become more difficult to read and understand due to the changes made by many Congresses.[94] Quantifying what has actually changed with each new edition could be useful for understanding how the law has evolved and as a first step for understanding how the law might be designed better. For example, measuring changes reveals what sections have been modified by Congress frequently and what sections have withstood the test of time.

For large software codebases, because many programmers are working on the code at the same time, mechanisms that ensure that changes do not break functional code or create conflicts are needed. The practice of software engineering has adopted "version control systems" to handle these problems.[95] Given that the U.S. Code is the product of many individual members of Congress over time, similar mechanisms are needed. Thus, a software engineering-inspired version control approach to the law could be a reasonable future method of managing legislative changes.

For this metric, we focus on the goal of developing software engineering-inspired tools for visualizing and communicating changes to the Code. We quantify two types of changes: (1) the aggregate number of words added or deleted, and (2) the appearance and spread of words over time and to different titles of the U.S. Code:

1. *Addition-and-Deletion Metrics.* When working on a codebase, software engineers routinely add, delete, revise, reorder, or restructure LOCs. For the purpose of analyzing the existing U.S. Code, a key insight from software engineers

[94] *See* Daniel Katz & Michael James Bommarito II, Measuring the Complexity of the Law: The United States Code 5 (Aug. 1, 2013) (unpublished working paper) (on file with the Journal of Business & Technology Law), *available at* `http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2307352` (The U.S. Code "contains hundreds of thousands of provisions and tens of millions of words").

[95] Christopher Menegay, *Using Source Code Control in Team Foundation* , Microsoft Developer Network (Sept. 2005), `http://msdn.microsoft.com/en-us/library/ms364074%28v=vs.80%29.aspx` (Version control systems can "manage files through the development lifecycle, keeping track of which changes were made, who made them, when they were made, and why"). Popular version control systems include: Concurrent Versions System ("CVS"), Apache Subversion ("SVN"), Mercurial, and Git. *See Concurrent Versions System* , NONGNU.ORG, `http://www.nongnu.org/cvs/` (last visited Feb. 10, 2015); *Apache Subversion* , APACHE SOFTWARE FOUND., `https://subversion.apache.org/` (last visited Feb. 10, 2015); Mercurial SCM, http://mercurial.selenic.com/ (last visited Feb. 10, 2015); GIT-SCM.com, http://git-scm.com/ (last visited Feb. 10, 2015).

is how they communicate and visualize changes to a document. Specifically, although editing software code can involve many high-level thought processes, they can be communicated through two operations: the addition and deletion of LOCs. In software, such a comparison of two versions of the same document is called a "diff" operation.[96] Revising an existing lie of code is simply the deletion of the existing line and the addition of a new line. When a team member makes changes, the rest of the team can easily identify where changes were made to a document and what those changes were, essentially by viewing redline comparisons. Further, the number of lines changed may suggest whether the revision was large or small. The act of comparing two versions of a document is a fundamental operation that programmers use regularly.[97]

Each historical edition of the U.S. Code is a snapshot of the laws at one instance in time. Using text-matching techniques, we can also apply the "diff" concept to two versions of a legal document; the only difference is that, instead of computer instructions, the law is written in English. Text matching is simply the task of detecting whether a sequence of words in one version of a document exists in a previous version. When applied to an entire document, it is possible to calculate what percentage of a document is new and what percentage previously existed.

2. *Word-based metrics.* Software engineers also use a number of other text-based tools in their daily work routines. Similar to how Internet search engines help users find relevant documents online or "find files" programs help computer users locate documents on their own computer, software engineers might search for specific terms in a codebase or their frequency to help them do their work.[98] Looking for the existence of terms throughout a codebase might, for example, help the software engineer determine whether a feature has already been imple-

[96] *GNU Tools*, UNIXhelp for Users (Sept. 22, 1993), `http://unixhelp.ed.ac.uk/CGI/man-cgi?diff`.

[97] *Id.*

[98] *See How to: Programmatically Search for and Replace Text in Documents* , Microsoft Developer Network, `http://msdn.microsoft.com/en-us/library/f1f367bx.aspx` (last visited Jan. 31, 2015) (explaining how to use Microsoft Word's "find" function).

mented or assess the design conventions that the team has used. In principle, these search techniques could also be applied to snapshots of the codebase over time to identify changes.

Understanding changes in the law, of course, requires going beyond simple length measurements. It is interesting, for example, to detect the first appearance of particular words in the U.S. Code; given our historical dataset, doing so is quite straightforward. In addition, we count the number of times that each word appears in each edition of the U.S. Code. Similar efforts have been employed by Google to count the appearance of terms in all English literature,[99] the New York Times for its news coverage,[100] and by other researchers for U.S. Supreme Court opinions.[101] In our case, these measurements are useful because they reflect the extent to which the U.S. Code covers different concepts.

Coupling. The U.S. Code is not simply a long passage of text; the legislative drafting manuals of both houses of Congress state that individual sections of legislation should be organized into titles, sections, sub-sections, sub-clauses, and other subdivisions.[102] Moreover, these subdivisions often reference each other.[103] For instance, one part of the U.S. Code might refer to definitions in another part, creating dependencies between them. The coupling of various parts of the U.S. Code creates nonlinearities that can make the code more challenging to parse and revise. In particular, a reader must now explore different "pathways" of references to fully understand a certain domain of law. Furthermore, revisions to any part of a chain of references could contribute to unknown, unintended downstream effects. Mapping the large-

[99] *See* Ben Zimmer, *Google's Ngram Viewer Goes Wild* , Atlantic (Oct. 17, 2013, 9:17 AM), `http://www.theatlantic.com/technology/archive/2013/10/googles-ngram-viewer-goes-wild/280601/` .

[100] Alexis Lloyd, *Chronicle: Tracking New York Times Language Usage over Time*, N.Y. Times (July 23, 2014), `http://blog.nytlabs.com/2014/07/23/chronicle-tracking-new-york-times-language-use-over-time/`.

[101] Daniel Martin Katz et al., Legal N-Grams?: A Simple Approach to Track the "Evolution" of Legal Language (Dec. 13, 2011) (unpublished paper) (on file with the Journal of Business & Technology Law), *available at* `http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1971953` .

[102] *See* House Drafting Manual, *supra* note 78, at 23–24; Senate Drafting Manual, *supra* note 76, at 8–10.

[103] *See* Katz & Bommarito, *supra* note 94 (citing 11 U.S.C. §101 as an example of a statute that contains both "within-Title" references and "cross-Title" references).

scale structure of cross-references in the U.S. Code, therefore, may reveal potential vulnerabilities in the law.

1. *Modularity in Software.* In the software context, good software systems are easy to separate into different modules, with the interface between modules being kept relatively sparse and simple.[104] The notion of "modularity" is the central idea behind "object-oriented programming," which is a fundamental design pattern in programming large software systems today and the focus of many seminal computer science papers and textbooks.[105]

   Object-oriented programming has become a dominant paradigm in software because it leverages the power of abstraction and modularity.[106] For example, a powerful word processor application like Microsoft Word has many functions, including formatting, citation management, checking spelling and grammar, and document printing options.[107] To manage this complexity, large software systems are split into modular subsystems. Smaller, more agile teams of software engineers are responsible for each of these modules, and each team only needs to understand the input-output behavior of other modules with which its module interacts.[108] Object-oriented design, therefore, leads to clearer lines of responsibility, both from a software standpoint and a human team management standpoint. This modularity results in more efficient coding, debugging, and, ultimately, more robust software.

   One way to study modularity is to interpret the system as a network (also known

[104] McConnell, *supra* note 85, at 38.

[105] Ola Berge et al., Learning Object-Oriented Programming (Nov. 23, 2007) (unpublished paper) (on file with the Journal of Business & Technology Law), *available at* `https://telearn.archives-ouvertes.fr/file/index/docid/190184/filename/Berge_2003.pdf` .

[106] *See, e.g.* , Leslie Kaelbling et al., *Introduction to Electrical Engineering and Computer Science I: Syllabus* , MIT OpenCourseWare, `http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-01sc-introduction-to-electrical-engineering-and-computer-science-i-spring-2011/Syllabus/` (last visited Jan. 21, 2015) (setting a goal to teach "the fundamental design principles of modularity and abstraction in a variety of contexts from electrical engineering and computer science").

[107] *See Word Object Model Overview* , Microsoft Developer Network, `http://msdn.microsoft.com/en-us/library/kw65a0we.aspx` (last visited Feb. 10, 2015).

[108] McConnell, *supra* note 85, at 21–22.

as a graph), where each function or variable corresponds to a node in the network, and there is an "edge" (a connection) from component A to component B if component B references component A.[109] A rich body of algorithms and techniques have been developed to characterize the properties of these networks.[110] Continuing with the Microsoft Word example, when a user decides to print a document, the "user interface" module connects to the "print" module.[111] Any major software system involves multiple references among its component modules; good object-oriented design suggests that cross-references should be used only when they are necessary, to avoid needless dependencies and complexity.

For any given software codebase, it is possible to construct and analyze its nodes and edges in aggregate. The resulting network structure can provide insights into the nature of the software system, such as how robust it is and where its vulnerabilities likely reside. The network map can also provide a sense of the different categories of modules that exist in a software system. Previous work, for example, has examined the core-periphery architecture common to many large software systems.[112] The portion of the network to which a certain module belongs can provide information about how the module relates to the rest of the system.[113]

2. *Modularity in the U.S. Code.* The same modularity principles can be applied to the law. We can interpret each section of the U.S. Code as a node of the network, with citations to sections as the network's edges. We can then analyze the graph structure for novel insights into the structure of the U.S. Code.

   As a concrete example, 37 U.S.C. §329, which describes an incentive bonus for

[109] Uday P. Khedker et al., Data Flow Analysis 234 (2009).

[110] *See, e.g.* , Thomas H. Cormen et al., Introduction to Algorithms 587 (3d ed. 2009) (describing how computer scientists can use algorithms and graphing techniques to solve computational problems).

[111] *See* Beth Melton, *Organizing Your Macros* , Microsoft Word MVP (Nov. 1, 2002, 9:52 PM), `http://word.mvps.org/faqs/macrosvba/OrganizeMacros.pdf` .

[112] Alan MacCormack et al., The Architecture of Complex Systems: Do Core-Periphery Structures Dominate? 1 (Jan. 19, 2010) (unpublished paper) (on file with author), *available at* `http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1539115` .

[113] *Id.* at 7.

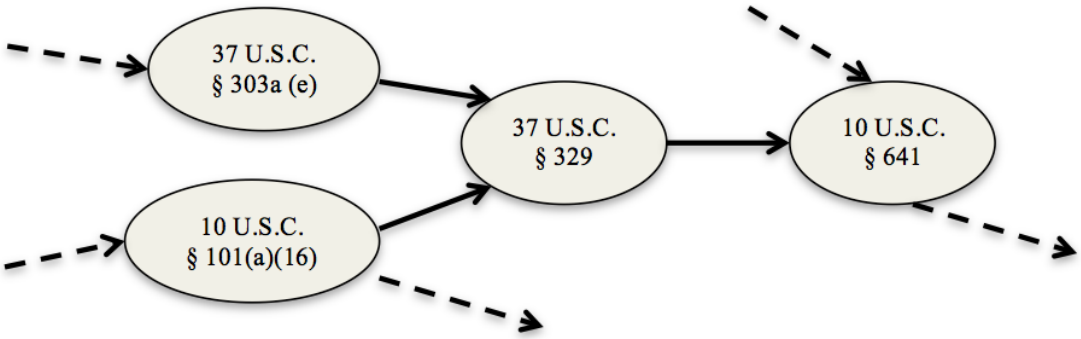


Figure 3-1: Network representation of references to and from 37 U.S.C. §329

retired or former members of the military, cites exactly two other sections, 37 U.S.C. §303a(e) (general provisions of special pay in the military), and 10 U.S.C. § 101(a)(16) (a definition of "congressional defense committees").[114] Meanwhile, 37 U.S.C. § 329 is cited by one other section, 10 U.S.C. §641, which notes that other laws in Title 10 of the U.S. Code do not apply to the officers to whom the bonus in 37 U.S.C. §329 applies.[115] Locally, the network is shown in Figure 3-1 (with dashed arrows representing links to and from other parts of the U.S. Code).

This simple representation immediately shows a chain of citations in which modifying 37 U.S.C. §303a(e) could have ramifications for 10 U.S.C. §641.[116] Now, imagine a longer chain with multiple branches, some of which could refer back to the section being modified. These chains can be used to identify complex sequences of legal implications that even the most knowledgeable and intelligent human cannot fully comprehend without technological assistance.

The entire U.S. Code comprises a large network with many references. This network can be analyzed in many ways; previous work, for instance, has sought to identify important U.S. Code sections by following references and determining which sections are encountered most often.[117] In our work, we examine the U.S.

[114] *See* 37 U.S.C. §§303(a), 329 (2012); 10 U.S.C. §101 (2012).
[115] *See* 37 U.S.C. §329; 10 U.S.C. §641 (2012).
[116] *See supra* Figure 3-1.
[117] *See, e.g.,* Katz & Bommarito, *supra* note 94, at 1, 6.

Code network in the following three ways.

First, for the historical dataset, we examine how sections from bills passed by Congress map to sections in the U.S. Code.[118] This data is available from the OLRC for every bill ever passed by Congress (including, interestingly, public laws before the U.S. Code came into being in 1926).[119] Specifically, for selected recent legislation, we find previously enacted laws that have the most overlapping number affected U.S. Code sections. This method allows us to find groups of similar laws by domain. For instance, the laws most similar to the Dodd-Frank Wall Street Reform and Consumer Protection Act tend to be finance and banking laws.[120]

Second, for the current law, we apply concepts from recent work on the network architecture of software codebases to describe the structure of U.S. Code titles and selected bills passed by Congress.[121] This analysis is based on finding the network's "core," which is the largest interconnected collection of nodes in the network.[122] More precisely, we define the core as the largest "strongly connected" component of the network.[123]

Third, we identify important sections by using the structure of the network of cross-references in the current U.S. Code.[124] Specifically, we use a link analysis

[118] *About the Table III Tool* , Office of the Law Revision Counsel: U.S. Code, `http://uscode.house.gov/table3/table3explanation.htm` (last visited Jan. 22, 2015). The Office of the Law Revision Counsel provides tables that "show where recently enacted laws will appear in the United States Code and which sections of the Code have been amended by those laws." *See United States Code Classification Tables* , Office of the Law Revision Counsel: U.S. Code, `http://uscode.house.gov/classification/tables.shtml` (last visited Jan. 22, 2015).

[119] *Table III Tool* , Office of the Law Revision Counsel: U.S. Code, `http://uscode.house.gov/table3/table3years.htm` (last visited Jan. 22, 2015).

[120] *See infra* Part IV.C and notes 168–69.

[121] *See, e.g.* , Carliss Baldwin et al., *Hidden Structure: Using Network Methods to Map System Architecture* (Harv. Bus. Sch. Working Paper, No. 13-093, May 2013), *available at* `http://dash.harvard.edu/handle/1/10646422` (describing an operational methodology to characterize complex technical system architecture); Alan Grosskurth & Michael W. Godfrey, Architecture and Evolution of the Modern Web Browser 1–2, 5, 18 (June 20, 2006) (unpublished paper) (on file with author), *available at* `http://plg.uwaterloo.ca/~migod/papers/2006/jss-browserRefArch.pdf` (presenting a reference architecture for web browsers).

[122] *See* Baldwin et al., *supra* note 121, at 2, 8.

[123] *See infra* App. A (defining mathematical terms that appear in this Article's network analysis, including "core" and "strong connectedness").

[124] *See USLM User Guide* , *supra* note 83; *U.S. Code* , *supra* note 84.

algorithm very similar to PageRank, popularized by Google as their method of ranking the importance of individual webpages.[125] The idea of PageRank is that each section in the U.S. Code has references to and from other sections, and a section that has many references to it is likely more important.[126] Further, if an important section refers to other sections, those sections may also be important. Using this intuition, the relative importance of all sections in the U.S. Code can be calculated. Previous work has applied this approach to academic literature[127] and, in the domain of law, the social network of the U.S. law professoriate.[128]

Complexity. The law is riddled with conditional statements, exceptions, and special cases.[129] Applying different rules to different situations is not inherently bad; however, such "balancing tests" make it more challenging to fully appreciate the consequences of a given piece of legislation.[130] Further, an excessive number of conditional statements might suggest that the underlying rule is faulty, requiring many special cases and exceptions. For these reasons, methods to count the number of statements that exist in the law might be useful for analyzing the U.S. Code.

Analogously, software code often contains conditional statements of the form in Figure 3-2 . If a condition is met, then some subroutine A is executed, and if the condition is not met, some other subroutine B is executed.[131] Each time a conditional statement appears, the possible execution of the software forks into two paths. Fur-

[125] *See* Sergey Brin & Lawrence Page, *The Anatomy of a Large-Scale Hypertextual Web Search Engine* , 30 J. Computer Networks & ISDN Sys. 107, 109–10 (1998).

[126] *See id.* at 109–10, 117; *see also* Lawrence Page et al, The PageRank Citation Ranking: Bringing Order to the Web (1998), *available at* `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.1768` .

[127] Carl T. Bergstrom et al., *The Eigenfactor Metrics* , 28 J. Neuroscience 11433 (2008).

[128] Daniel M. Katz et al., *Reproduction of Hierarchy? A Social Network Analysis of the American Law Professoriate* , 61 J. Legal Educ. 76 (2011).

[129] *See, e.g.* , Restatement (Second) of Torts §§314, 314A (1965) (providing that a person has no duty to act when another person requires the first person's aid or protection, unless a special relationship exists between them); U.C.C. §2-207 (2002) (prescribing that a "definite and seasonable expression of acceptance" forms a contract even if the offeree's terms differ from the offeror's, unless the offeree expressly conditions acceptance on the offeror's assent to the different terms).

[130] Patrick M. McFadden, *The Balancing Test* , 29 B.C. L. Rev. 585, 636–42 (1988), *available at* `http://lawdigitalcommons.bc.edu/bclr/vol29/iss3/2/` .

[131] *See* McConnell, *supra* note 85, at 355–56, 358–59; *see also Conditional (Computer Programming)* , Wikipedia, `http://en.wikipedia.org/wiki/Conditional_(computer_programming)` (last visited Jan. 22, 2015).

```
IF (condition)
    (execute subroutine A)

ELSE
    (execute subroutine B)
```

Figure 3-2: if-else statement common in software code

ther, conditional statements can be nested (there can be conditional statements inside subroutines), which can lead into exponentially many possible execution paths for a given input.[132] The complexity that conditional statements introduce is formalized in software engineering as "cyclomatic complexity" (sometimes known as "McCabe's complexity"), which is the number of times a piece of code has to make a decision, *i.e.* , the number of paths in software.[133] It can be computed by assigning a score to each conditional statement that a piece of software encounters.[134]

To create an analogous metric for the U.S. Code, we count the number of conditional terms in a passage of text. We count the occurrences of the following conditional terms in a law or a section of the U.S. Code: "if," "except," "but," "provided," "when," "where," "whenever," "unless," "notwithstanding," "in no event," and "in the event."[135] This list is not exhaustive, and we do not expand these root words, but it provides an indication of the exceptions and special cases that are found throughout the U.S. Code. Section 308 of the Senate Legislative Drafting Manual, entitled "Conditional Provisions and Provisos," offers guidelines on what words to use: it recommends "if" instead of "when" or "where" to indicate a condition, and "except that," "but," or "if" instead of phrases involving the word "provided."[136] We include both the recommended and non-recommended terms because laws are not obligated to follow these guidelines—an online search of the current U.S. Code shows that all of these terms still exist in the U.S. Code to describe a conditional statement.[137]

[132] McConnell, *supra* note 85, at 499–500.

[133] Thomas J. McCabe, *A Complexity Measure* , 2 IEEE Transactions on Software Eng'g 308, 308–20 (1976).

[134] *Id.* at 308–10, 318–19.

[135] *See infra* Part IV.D.

[136] Senate Drafting Manual, *supra* note 76, at 69.

[137] *See Search the United States Code* , Office of the Law Revision Counsel: U.S. Code, `http://uscode.house.gov/search.xhtml` (last visited Jan. 22, 2015).

Table 3.1: Description of Principles and Metrics for U.S. Code

Principle	**Proposed Metrics (Evolution of U.S. Code)**	**Proposed Metrics (Current Laws and Titles)**
Conciseness: Good code should be as long as it needs to be, but no longer.	Change in total number of words	Total number of words
Change: Code that exhibits large or frequent changes may suggest defects. Large, untested changes can also produce new defects.	Number of words added or deleted; counts of specific words and terms versus time; first appearance of words in U.S. Code by title	N/A
Coupling: Modular code is more robust and easier to maintain than code with unnecessary cross-dependencies.	Bills affecting similar sections	Size of cross-reference "network core" versus "network periphery"; Google PageRank-inspired methods
Complexity: Code with a large number of conditions, cases, and exceptions is difficult to understand and prone to error.	Change in number of condition statements in code (cyclomatic complexity)	Total number of condition statements by section (cyclomatic complexity)

Summary. Table 3.1 summarizes the previous four sections. The first column provides brief definitions of these four principles, and the second and third columns identify metrics borrowed from the software engineering community for the historical and current datasets, respectively. In the remainder of this Article, we apply our metrics and show the results of our analyses and visualizations.

3.6 Evolution of the U.S. Code

To understand the evolution of the U.S. Code, we used the following datasets:

1. Historical U.S. Code texts under license from William S. Hein & Co., Inc.[138]

2. For certain comparisons, a more recent version of the U.S. Code from the OLRC.[139]

3. A document called "Table III," published by the OLRC, which shows, on a section-by-section basis, how an enrolled bill maps to the U.S. Code.[140]

Using our historical dataset, we analyzed and visualized changes to the U.S. Code since 1926. For each of the four software engineering principles listed in the previous section, we comment on insights that emerge from studying the U.S. Code through these metrics.

3.6.1 Conciseness: Evolution of the Size of the U.S. Code

Figure 3-3 is a stacked area graph of the size of the U.S. Code, organized by title and measured in the number of words. Consistent with the popular conception of federal laws, the size of the U.S. Code has grown continuously since 1926.[141] Moreover, the rate of growth is increasing.[142]

This simple length-based analysis also illustrates that different titles of the U.S. Code are different sizes. For example, Title 42 (Public Health and Welfare) is the longest.[143]

[138] *See U.S. Code* , *supra* note 84.
[139] *See Current Release Point* , *supra* note 82; *USLM User Guide* , *supra* note 83.
[140] *Table III Tool* , *supra* note 119. The authors have written software that parses data from the Table III Tool into machine-readable form for network-based analyses. *See uscode/table 3* , GitHub, `https://github.com/unitedstates/uscode/tree/master/table3` (last visited Jan. 22, 2015). This free software is available as part of the @unitedstates project. *See id.* An "enrolled bill" is "the final copy of a bill or joint resolution which has passed both chambers in identical form." *See Glossary* , U.S. Senate, `https://www.senate.gov/reference/glossary_term/enrolled_bill.htm` (last visited Jan. 22, 2015).
[141] *See* Figure 3-3
[142] *See* Figure 3-3
[143] *See* Figure 3-3

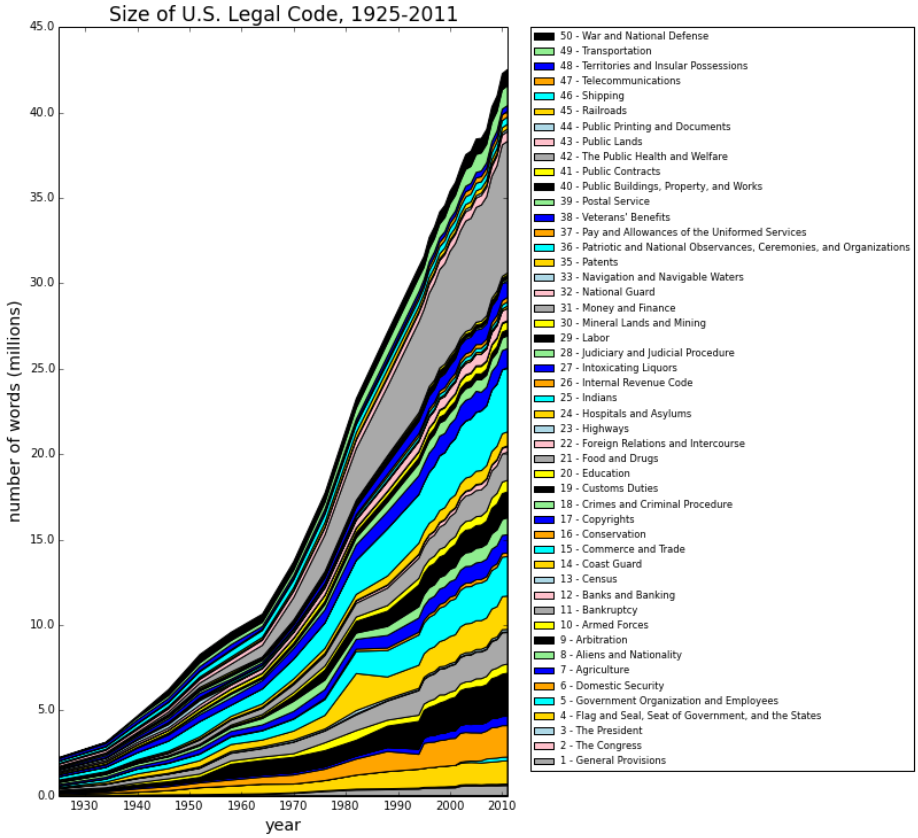


Figure 3-3: Number of words in the U.S. Code by title.

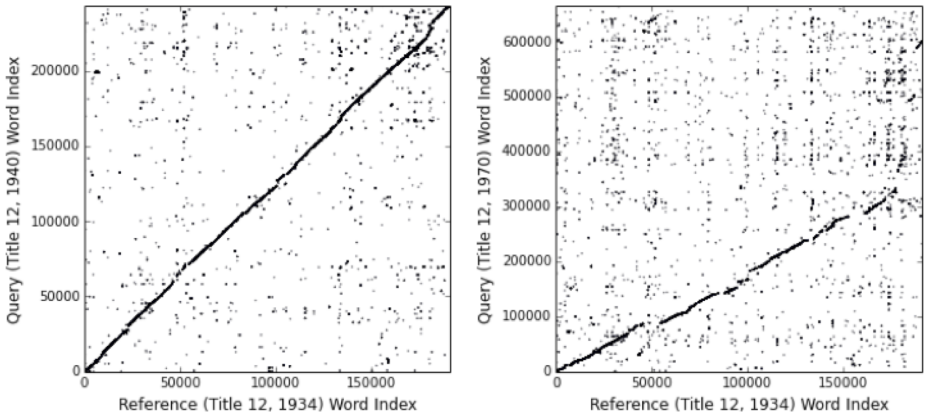


Figure 3-4: Title 12 (Banks and Banking) comparisons between 1934 and 1940 editions (left) and 1934 and 1970 editions (right).

3.6.2 Change: Evolution of Content in the U.S. Code

The inadequacy of assessing the U.S. Code by length alone is apparent when analyzing changes over time. For example, despite the increasing size of the U.S. Code over time, the length is actually the net result of numerous laws enacted and repealed. Thus, the "diff" function allows us to more accurately assess the quality and quantity of change. Moreover, the first appearance and frequency of terms in the U.S. Code are also informative for studying its evolution.

Addition and Deletion of Words. As described *supra* in Section 3.5, the U.S. Code is not neatly organized into individual lines like software, so we treat each title as a sequence of words to find matches between sequences. Figure 3-4 shows the output of the document comparison process. We chose Title 12 (Banks and Banking) to illustrate our approach. When there is a matching sequence of words, a black dot is drawn on the plot; when there is a mismatch, no dot is drawn. The following insights can be gained by examining the two plots:

The dark diagonal line from the bottom-left to the top-right of the left plot in Figure 3-4 indicates that the 1934 and 1940 versions of the Title 12 are largely the same. The relatively small breaks in this dark-blue diagonal line indicate there were relatively few changes between 1934 and 1940. In contrast, the diagonal line is much less intact in the comparison between the 1934 and 1970 versions of the U.S. Code.

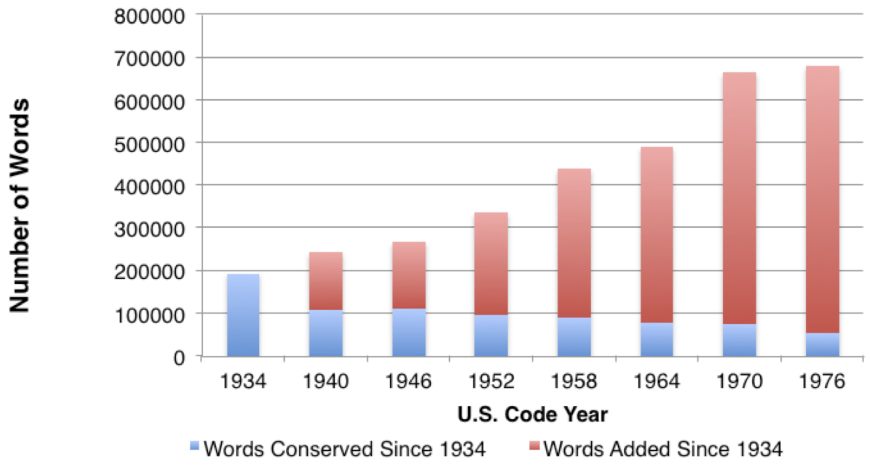


Figure 3-5: Words conserved and added to Title 12 between 1934 and 1976

This pattern indicates that there are large differences between the two documents; that is, there were far more changes between 1934 and 1970 than there were between 1934 and 1940. In particular, large amounts of text were added to the end of Title 12 sometime prior to 1970.

Using the text comparison technique shown above, we can go beyond simply counting the number of words and determine how many words were added and deleted with each subsequent edition of the U.S. Code. Figure 3-5 summarizes these changes between 1934 and 1976 for Title 12. For graphing purposes, instead of showing "words deleted," we show "words conserved" in order for the stacked bar graph to show the total number of words in each edition of Title 12. This graph illustrates that the length changes in Title 12 are the product of both the addition of new passages of text and the deletion of passages of text that existed in 1934, though the vast majority of the length comes from words added.

Term Frequency Counts. As a first step toward understanding the content of the U.S. Code, we built a U.S. Code "term-count viewer." Figure 3-6 presents some screenshots of our tool to count the frequency of terms by year. In this figure, a) illustrates the rise of legislation related to the telephone, and the slow decline of the telegraph;[144] b) shows how "homeland security" entered the discourse between

[144] *See* Derek Thompson, *The 100-Year March of Technology in 1 Graph* , Atlantic (Apr. 7, 2012, 1:08 PM), `http://www.theatlantic.com/technology/archive/2012/04/the-100-year-march-of-technology-in-1-graph/255573/` (graphing telephone penetration

2000 and 2006, after 9/11;[145] and c) corresponds to the invention of the credit card and laws related to consumer protection in the 1960s and beyond.[146] These term frequency plots illustrate the attention that legislators and society devoted to new domains of law in different decades.

First Appearance of Words. Along with examples of term frequency patterns, we can also examine when words first appeared in the U.S. Code. Table 3.2 shows new terms that appeared in each edition of the U.S. Code between 1952 and 2006. The top 10 words in terms of their total count in the 2006 edition of the U.S. Code is shown, in order to show words that first appeared in a given year and have now become commonplace in the U.S. Code. For example, in our dataset, the term "television" first appears in the 1952 edition of the U.S. Code and can be found 1297 times.[147]

Some terms (such as "Palau" and "Mariana") reflect routine bookkeeping changes to the U.S. Code, such as changes corresponding to entities that signed Compacts of Free Association with the United States.[148] The timing of other words, such as "television," "telecommunications," "pesticide," or "privacy," reflects when these concepts and entities first received the attention of federal law. Meanwhile, other terms reflect a change in language usage: the appearance of the term "servicemember(s)" indicates a move away from gender-specific terms.[149]

---

from 1900–2005).

[145]The word "homeland" appears less than twenty times in U.S.C. (2000), but more than 1,700 times in U.S.C. (2006). *Search the United States Code* , *supra* note 137.

[146]Although revolving debt credit cards first appeared in the 1950s, significant regulation did not occur until the 1960s. *See* John T. Finley, *Consumer (Bankcard) Debt and Regulation—Are Things Working?* , 17 Proc. Acad. Legal, Ethical & Reg. Issues 7 (2013), *available at* `http://www.alliedacademies.org/public/proceedings/Proceedings32/ALERI%20Proceedings%20Spring%202013.pdf` .

[147]*See* U.S.C. (1952); *e.g.* 18 U.S.C. §1343 ("Whoever, having devised or intending to devise any scheme or artifice to defraud, . . . transmits or causes to be transmitted by means of interstate wire, radio, or *television* communication . . . ." (emphasis added)); 26 U.S.C. §3403(c) ("Parts or accessories (other than tires and inner tubes and other than radio and *television* receiving sets) for any of the articles enumerated in subsection (a) or (b), 8 per centum, except that on and after April 1, 1954, the rate shall be 5 per centum." (emphasis added)).

[148]Compact of Free Association, U.S.-Marshall Islands, Apr. 30, 2003. T.I.A.S. No. 04-501.

[149]*See, e.g.* , National Defense Authorization Act for Fiscal Year 2012, Pub. L. No. 112-81, §631(a), 125 Stat. 1298, 1452 (2011) ("Recognizing the complexities and the changing nature of travel, the amendments made by this section provide the Secretary of Defense and the other administering Secretaries with the authority to prescribe and implement travel and transportation policy that is simple, clear, efficient, and flexible, and that meets mission and *servicemember* needs. . . ." (emphasis added)); Servicemembers Civil Relief Act, Pub. L. No. 108-189, 117 Stat. 2835 (2003).

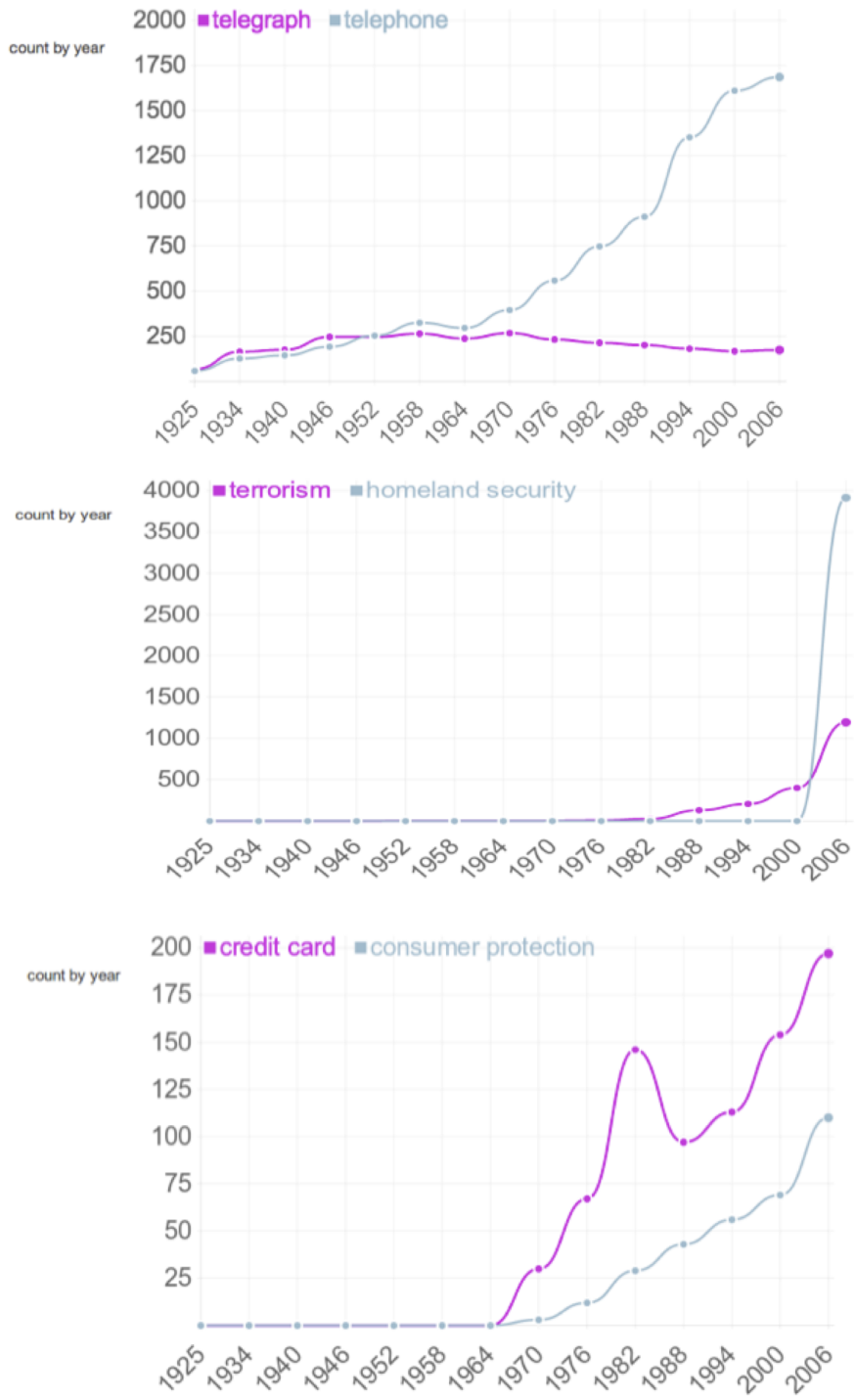


Figure 3-6: Term frequency plots over time for selected phrases: a) "telegraph" vs. telephone"; b) "terrorism vs. homeland security"; c) "credit card vs. consumer protection"

Table 3.2: First Appearance of Terms in the U.S. Code

1952	1958	1964	1970	1976
television (1297) operational (1210) pipeline (1199) terrorism (1193) workforce (1159) telecommunications (1068) victim (1013) reconciliation (975) satellite (912) significantly (823)	infrastructure (1341) rulemaking (1062) pesticide (918) enhancement (915) micronesia (830) inpatient (829) elderly (791) confidentiality (716) statewide (700) global (685)	environmental (5811) guidelines (3477) technologies (2111) providers (1859) computer (1341) mariana (1318) environ (1150) monitor (1099) evaluations (907) privacy (788)	medicare (2553) subclause (1598) expertise (1149) strategies (964) outreach (889) ensuring (781) innovative (741) oceanic (722) affordable (719)	initiatives (684) chairperson (1112) terrorist (865) medicaid (683) update (615) digital (528) methodology (524) software (459) amtrak (455) syndrome (432) underserved (377)
1982	**1988**	**1994**	**2000**	**2006**
palau (566)[150] targeted (454) saharan (415) assistive (330)[151] swap (316) hospice (300) competitiveness (289) nonattainment (272) fueled (267) nonproliferation (266)	database (397) affordability (206) remic (194)[152] kg (188) privatization (177) servicemembers (163) alzheimer's (154) mammography (148) forensic (142) noncustodial (135)	internet (754) nafta (504) stalking (370) geospatial (232) mentoring (210) biodiesel (160) nonoriginating (151) databases (148) empowerment (148) countervailable (145)	tricare (331)[153] website (199) y2k (127) biobased (126) hubzone (112)[154] bliley (108)[155] vento (103) [156] telehealth (100) hass (93) cbtpa (83)[157]	servicemember (161) pdp (153)[158] cafta (137)[159] darfur (100) restyling (94) nanotechnology (77) safetea (75)[160] katrina (67) pandemic (63) atpdea (61)[161]

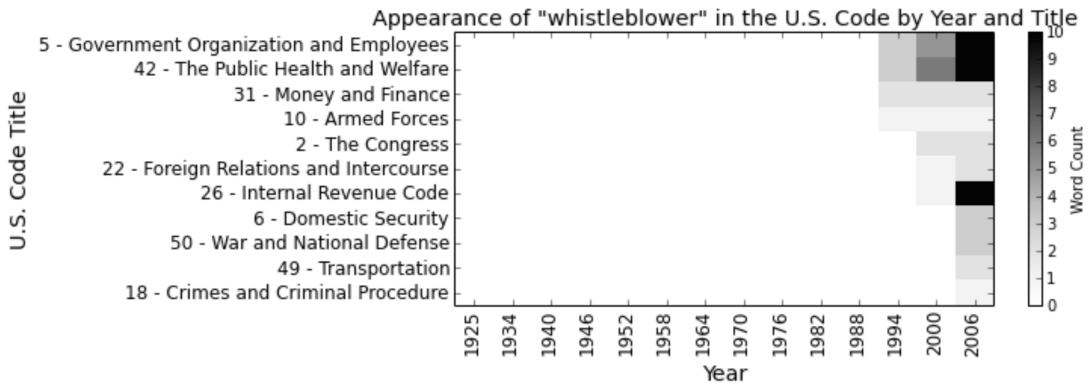


Figure 3-7: Appearance of "whistleblower" in U.S. Code titles by year and title

Trajectories of Terms in U.S. Code Titles. Finally, instead of merely listing the appearance of new terms in the U.S. Code, we can also examine the contexts in which they are used. In particular, our historical dataset makes it possible to track terms of interest across different titles of the U.S. Code. In Figure 3-7 , we show that the term "whistleblower" first appeared in Titles 5 (Government Organization and Employees), 42 (Public Health and Welfare), 31 (Money and Finance), and 10 (Armed Forces) in 1994.[162] It now is mentioned in a total of 11 U.S. Code titles. Meanwhile, as shown in Figure 3-8 , "privacy" is mentioned throughout the U.S. Code. Interestingly, its first appearance was in 1964 in Title 39 (Postal Service).[163] This visualization reveals when discourse framed around whistleblowers or privacy entered different titles of the U.S. Code.

3.6.3 Coupling: Evolution of Structure of U.S. Code

When Congress passes a bill and the President signs it into law, the OLRC incorporates the new law into the U.S. Code.[164] The OLRC keeps an online record of the mapping of every bill section to its corresponding section in the U.S. Code.[165] This mapping of bill sections to U.S. Code sections forms a network connection map (a

[162] *See Search the United States Code* , *supra* note 137.
[163] Privacy of Accounts, 39 U.S.C. §5212 (1964).
[164] *About Classification of Laws to the United States Code* , *supra* note 65.
[165] *See Table III Tool* ,*supra* note 119.

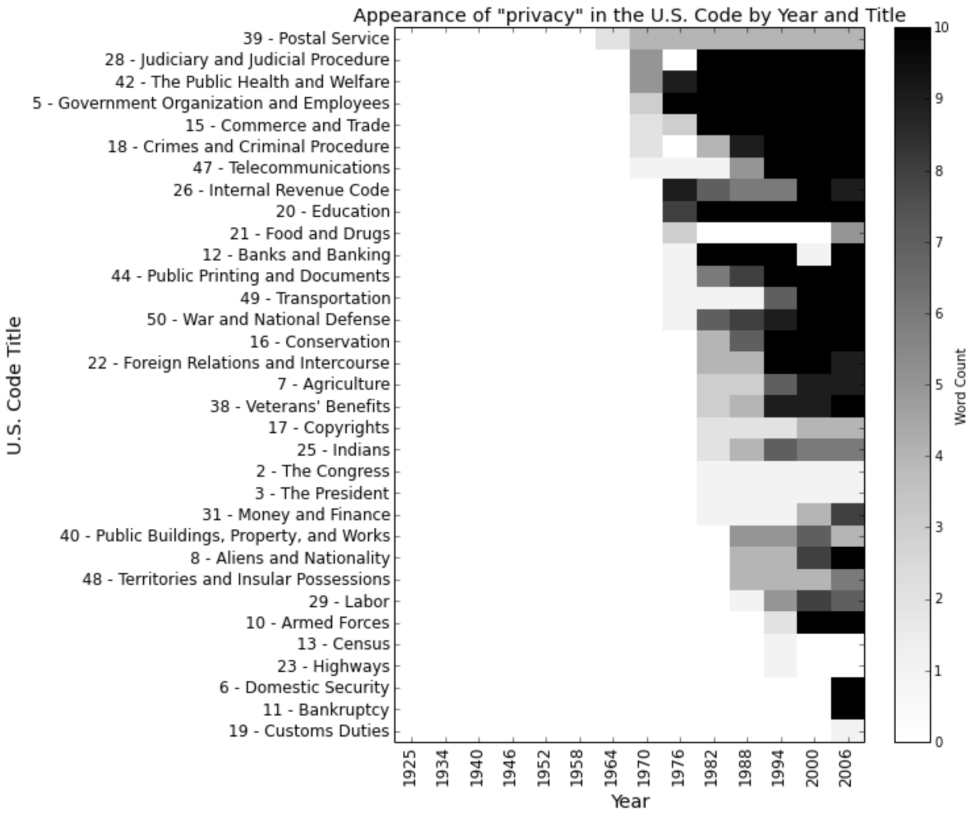


Figure 3-8: Appearance of "privacy" in U.S. Code titles by year and title

"graph" in computer science terms).

One application of this graph is to determine similar bills in terms of the overlap of the U.S. Code sections that they affect. If newly enacted laws are like new additions to software, then we can determine quantitatively which existing laws were changed or impacted most. We use the Jaccard similarity,[166] a mathematical measure of overlap of sets of entities, to calculate how similar two laws are in terms of sections affected: two bills that affect the exact same U.S. Code sections would have a Jaccard similarity of 1.0, while two bills that affect completely different sets of sections would have a Jaccard similarity of 0.0.[167]

As an illustration of this method, Table 3.3 shows the most similar bills to the Dodd-Frank Wall Street Reform and Consumer Protection Act ("Dodd-Frank") while Table 3.4 shows similar bills to the Patient Protection and Affordable Care Act (PPACA).[168] Notably, the bill ranked most similar to Dodd-Frank is Public Law 101-73 (the Financial Institutions Reform, Recovery, and Enforcement Act of 1989), which was the legislative response to the Savings and Loan Crisis in the late 1980s.[169] The list also contains other landmark pieces of legislation related to the financial sector at different points in the 20th century.

We can also visualize this similarity. Figure 3-9 shows show the sections of the U.S. Code affected by Dodd-Frank and similar laws, while Figure 9 shows laws similar to PPACA. For each of the bills, each dot represents a section of the U.S. Code. These dots are ordered by U.S. Code section number. Because these bills have a very large number of sections, they need to be shown in multiple rows. Only sections affected by at least one of the bills are represented, and the notations on the side indicate the sections corresponding to the first and last dots on each row. Stacked dots in the

[166] *See* Sheetal A. Takale & Sushma S. Nandgaonkar, *Measuring Semantic Similarity Between Words Using Web Documents* , Int'l J. Advanced Computer Sci. & Applications, Oct. 2010, at 78, 82; R. Real, *Tables of Significant Values of Jaccard's Index of Similarity* , 22 Miscellania Zoologica 29, 30 (1999).

[167] *See* Takale & Sushma, *supra* note 166, at 82; Real, *supra* note 166, at 30.

[168] *See* Dodd-Frank Wall Street Reform and Consumer Protection Act, Pub. L. No. 111-203, 124 Stat. 1376 (2010); Patient Protection and Affordable Care Act, Pub. L. No. 111-148, 124 Stat. 119 (2010).

[169] *See* Paul T. Clark et al., *Regulation of Savings Associations Under the Financial Institutions Reform, Recovery, and Enforcement Act of 1989* , 45 Bus. Law. 1013, 1013 (1990).

Table 3.3: Bills with Highest Similarity to Dodd-Frank Wall Street Reform and Consumer Protection Act

Rank (by Jaccard Similarity)	Public Law No.	Bill Name	Number of Sections in Bill	Jaccard Similarity
1	101-73	Financial Institutions Reform, Recovery, and Enforcement Act of 1989	284	0.113
2	90-321	Consumer Credit Protection Act	188	0.112
3	73-291	Securities Exchange Act of 1934	87	0.071
4	102-242	Federal Deposit Insurance Corporation Improvement Act of 1991	173	0.071
5	103-325	Riegle Community Development and Regulatory Improvement Act of 1994	245	0.051
6	95-630	Financial Institutions Regulatory and Interest Rate Control Act of 1978	166	0.051
7	96-221	Depository Institutions Deregulation and Monetary Control Act of 1980	125	0.050
8	106-102	Gramm-Leach-Bliley Act	155	0.049
9	102-550	Housing and Community Development Act of 1992	558	0.049
10	100-181	Securities and Exchange Commission Authorization Act of 1987	63	0.049

Table 3.4: Bills with Highest Similarity to Patient Protection and Affordable Care Act

Rank (by Jaccard Similarity)	Public Law No.	Bill Name	Number of Sections in Bill	Jaccard Similarity
1	74-271	Social Security Act of 1935	538	0.129
2	108-173	Medicare Prescription Drug, Improvement, and Modernization Act of 2003	234	0.122
3	94-437	Indian Health Care Improvement Act	156	0.120
4	78-410	Public Health Service Act of 1944	1227	0.103
5	105-33	Balanced Budget Act of 1997	424	0.086
6	102-573	Indian Health Amendments of 1992	141	0.081
7	111-152	Health Care and Education Reconciliation Act of 2010	116	0.074
8	110-275	Medicare Improvements for Patients and Providers Act of 2008	84	0.071
9	101-239	Omnibus Budget Reconciliation Act of 1989	706	0.062
10	101-508	Omnibus Budget Reconciliation Act of 1990	938	0.060

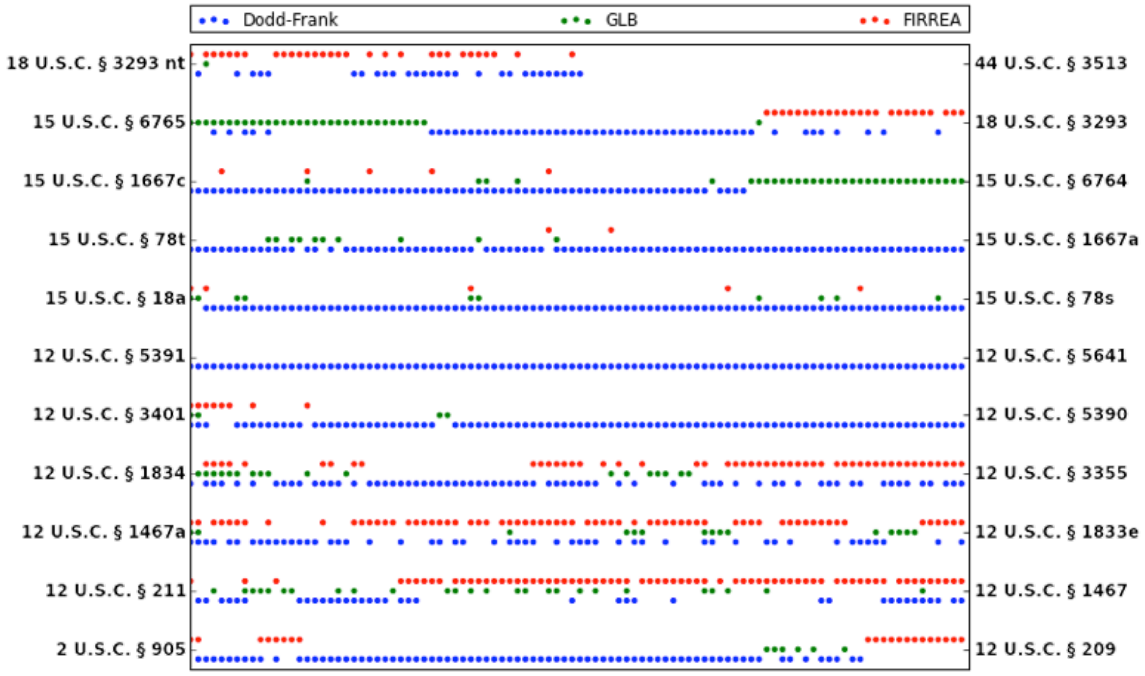


Figure 3-9: Comparisons of Sections of the U.S. Code affected by Dodd-Frank Wall Street Reform and Consumer Protection Act, Financial Institutions Reform, Recovery, and Enforcement Act of 1989 (FIRREA), and Gramm-Leach-Bliley Act (GLB)

same row indicate multiple bills affected those sections. For example, in Figure 9, as shown by the annotations, all three bills affected 42 U.S.C §1395yy, but only PPACA affected sections in 42 U.S.C. §280.[170] In the case of both PPACA and Dodd-Frank, it is worth noting that these laws, in addition to amending many existing sections related to other key bills, also created entirely new sections in the U.S. Code, which may explain why they did not overlap more with previous bills.[171]

[170] 42 U.S.C. §§280, 1395yy (2012).

[171] *E.g.* , Dodd-Frank Wall Street Reform and Consumer Protection Act §748 (codified at 7 U.S.C. §26 (2012)) (setting rewards for whistleblowers); Patient Protection and Affordable Care Act §3021(a) (codified at 42 U.S.C. §300jj–351 (2012)) (establishing the Center for Medicare and Medicaid Innovation).

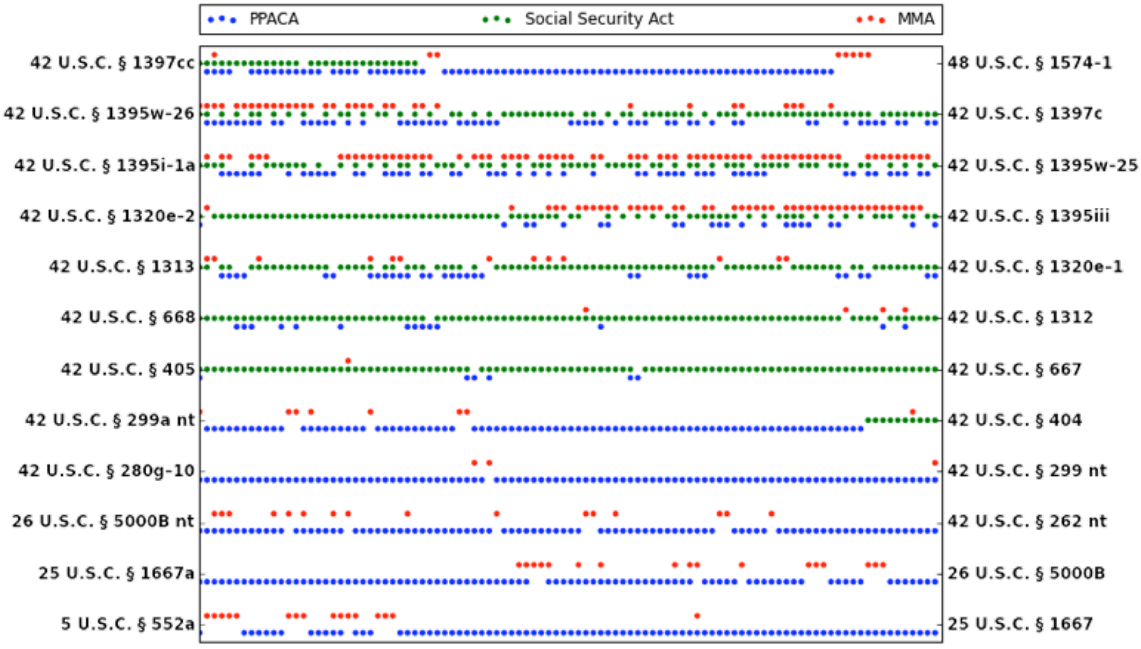


Figure 3-10: Comparison of sections of the U.S. Code affected by Patient Protection and Affordable Care Act (PPACA), Social Security Act of 1935, and Medicare Prescription Drug, Improvement, and Modernization Act of 2003 (MMA).

3.6.4 Complexity: Complexity: Evolution of Conditional Statements in the U.S. Code

Similar to measuring length, we can count the number of conditional statements by title in the U.S. Code over time. The results are shown in Figure 3-11. As with the length measurement, the number of conditional statements has also grown substantially over time. In the next two sections of this Article, we identify and explore titles and specific laws with particularly high cyclomatic complexity, which may be parts of the U.S. Code that are particularly difficult to understand.

3.7 Structure of Current Laws: 111th Congress

This section examines laws passed by the 111th Congress to determine whether our software engineering approaches can help identify the most complex laws that may be, consequently, prone to unintended consequences. The 111th Congress spanned

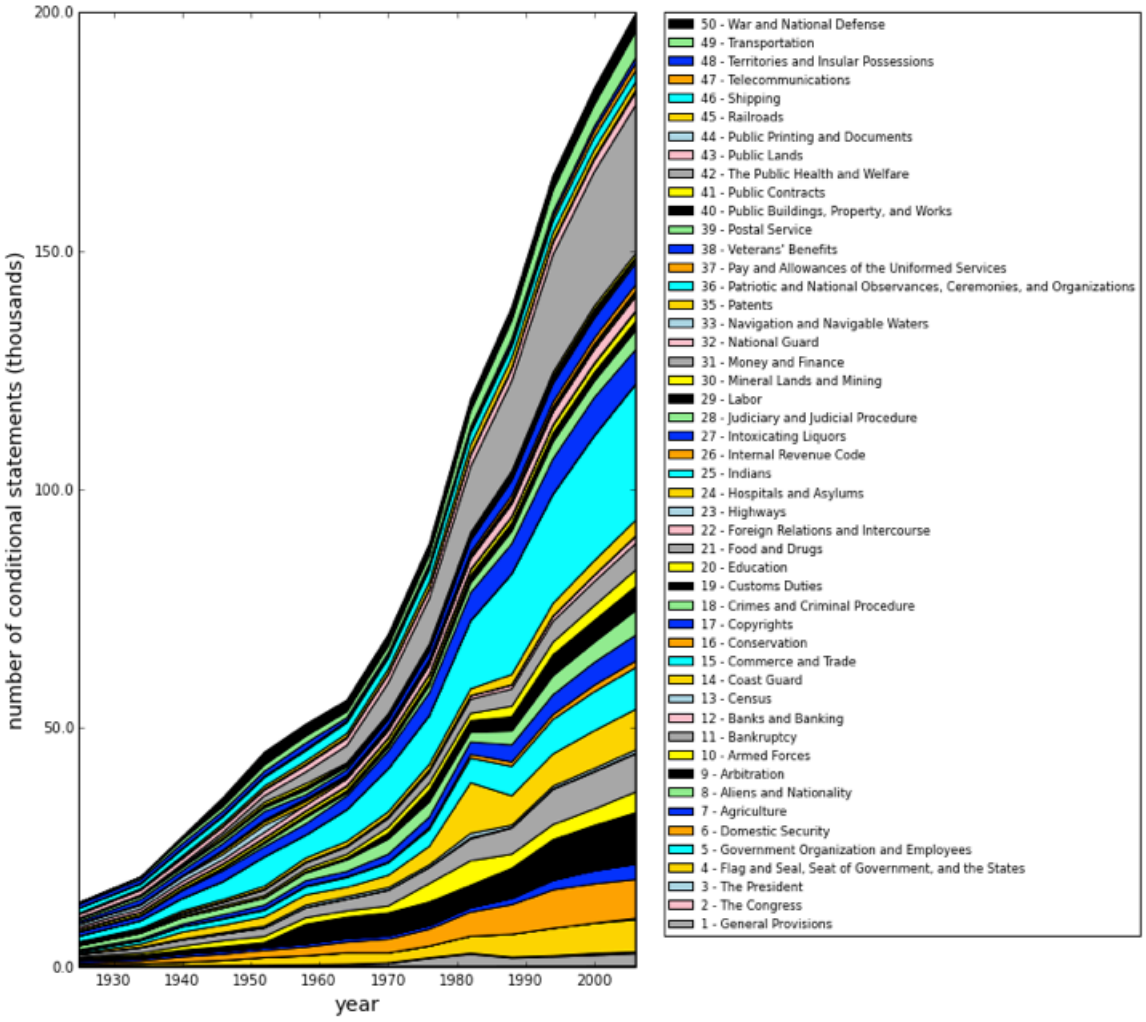


Figure 3-11: Cyclomatic complexity (number of conditional statements) in U.S. Code

the period from January 3, 2009 to January 3, 2011.[172] Some notable laws that it passed included Public Law 111-5, the American Recovery and Reinvestment Act of 2009 ("ARRA," or, informally, the "stimulus" bill);[173] Public Law 111-148, the Patient Protection and Affordable Care Act ("PPACA," or, informally, "Obamacare");[174] and Public Law 111-203, the Dodd-Frank Wall Street Reform and Consumer Protection Act (informally, "Dodd-Frank").[175] Our goal in this section and the next is to measure quantitatively the complexity of these laws, and use these measures to identify the effect that these laws had on the overall complexity of the U.S. Code.[176]

Our main results lead to three conclusions. First, laws that would be classified as "complex" or "important" by a human reader, such as PPACA or Dodd-Frank, are also very complex according to our software metrics.[177] Second, the average law is not very complex according to our measures.[178] Combined with our first point, this implies that there is a level of agreement between our techniques to identify complex laws and our findings with PPACA or Dodd-Frank. Third, our coupling metric helps identify two categories of "lengthy laws." The first type is appropriations acts, which are very long but do not have a high degree of coupling with the U.S. Code.[179] The second type includes laws such as PPACA, ARRA, or the extension of the Bush-era tax cuts in 2010, which show a high degree of coupling with the U.S. Code.[180] This

[172] *Past Days in Session of the U.S.* Congress, Congress.gov, https://congress.gov/past-days-in-session (last visited Jan. 22, 2015).

[173] American Reinvestment and Recovery Act of 2009, Pub. L. No. 111-5, 123 Stat. 115 (2009).

[174] Patient Protection and Affordable Care Act, Pub. L. No. 111-148, 124 Stat. 119 (2010).

[175] Dodd-Frank Wall Street Reform and Consumer Protection Act, Pub. L. No. 111-203, 124 Stat. 1376 (2010).

[176] Later in this chapter, we study the years 1995–2012 and show that, while the laws passed by the 111th Congress are complex, they are not uniquely so. *See infra* Part VI. No correlation exists between complexity and the party that controls Congress or the Presidency, and no pattern associates complexity to the presence or absence of a gridlocked government. *See infra* Apps. B, C. To the contrary, most complex laws seem to correspond to the 104th Congress (1995–97), which is well known for its disagreements between the executive and legislative branches, including a government shutdown. *See 1995–96 Government Shutdown* , Bancroft Libr., http://bancroft.berkeley.edu/ROHO/projects/debt/governmentshutdown.html (last updated Oct. 2, 2013);*infra* Apps. B, C.

[177] Patient Protection and Affordable Care Act; Dodd-Frank Wall Street Reform and Consumer Protection Act.

[178] *See* discussion *infra* Part VI.C. and in the appendix of this chapter.

[179] *See* discussion *infra* Part VII.

[180] *See infra* Tables 3.5 and 3.7

Table 3.5: Laws from the 111th Congress Ranked by Length

Public Law Number	Popular Name	Length (number of words)
111-11	Omnibus Public Land Management Act	191,864
111-8	Omnibus Appropriations Act	216,534
111-84	National Defense Authorization Act	274,329
111-203	Dodd-Frank	364,844
111-148	PPACA	384,324

coupling suggests that the content of these laws are more embedded in the "core" of the U.S. Code. Thus, our coupling measure can help quantify the extent to which laws have a more fundamental, structural effect on the U.S. Code.

While we focus on laws enacted by Congress, it is important to highlight that our techniques can be used in the future to analyze proposed laws. Our measure of coupling can give insights on how a proposed law will affect the rest of the U.S. Code. Our other measures can be used to compare two versions of a bill, and identify which sections of a bill can or should be simplified.

We show the top five laws passed by the 111th Congress according to length (Table 3.5 , coupling (Table 3.6), and complexity (Table 3.7). The results confirm, in a quantitative way, the intuition that laws such as Omnibus Appropriations Act, PPACA, and Dodd-Frank are complex.

It is reasonable to ask whether such complexity is significant. How much of an outlier are these particular laws from an average law enacted during the 111th Congress? This question is answered by examining the distributions of length (Figure 3-12), coupling (Figure 3-13), and complexity (Figure 3-14), which show the distributions of our metrics. These distributions are very thin-tailed, implying that the occurrence of these highly ranked laws is very low. Indeed, most laws have much lower values of these metrics.

One interesting observation is that the Omnibus Appropriations Act appears

Table 3.6: Laws from the 111th Congress Ranked by Coupling. The coupling metric used is the number of sections in the law that also belong to the core of the U.S. Code.

Public Law Number	Popular Name	Number of Sections in Core of U.S. Code
111-84	National Defense Authorization Act	143
111-312	Tax Relief, Unemployment Insurance Reauthorization and Job Creation Act	199
111-203	Dodd-Frank	232
111-148	PPACA	251
111-5	Stimulus Act	293

Table 3.7: Laws from the 111th Congress Ranked by Cyclomatic Complexity

Public Law Number	Popular Name	Cyclomatic Complexity
111-5	Stimulus Act	805
111-117	Consolidated Appropriations Act	1130
111-148	PPACA	1225
111-203	Dodd-Frank	1384
111-8	Omnibus Appropriations Act	1414

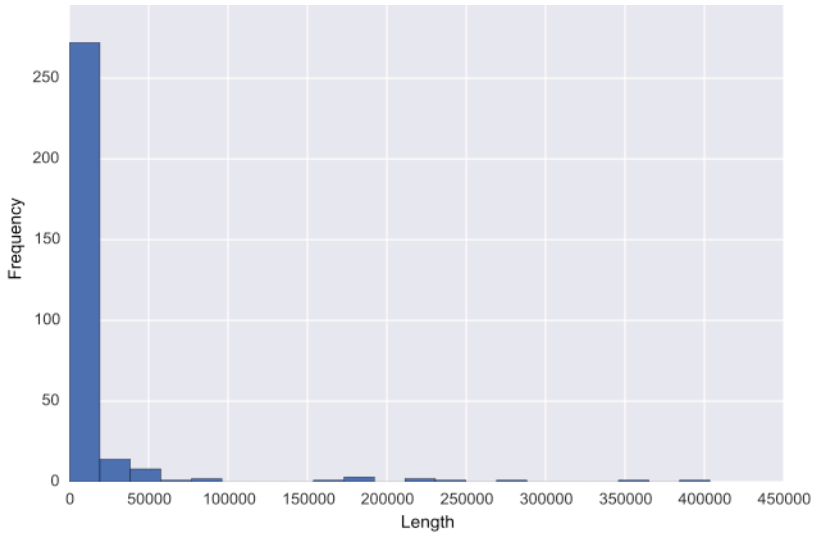


Figure 3-12: Distribution of lengths of laws passed by 111th Congress

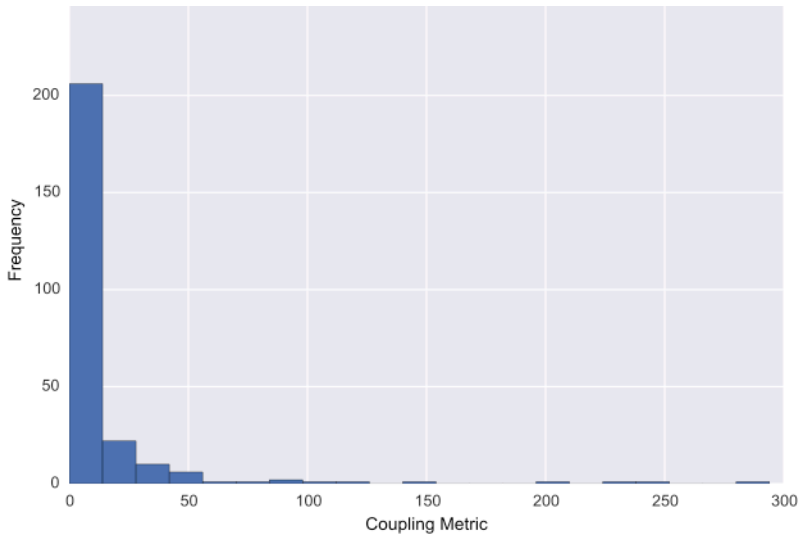


Figure 3-13: Distribution of coupling metric for laws passed by 111th Congress

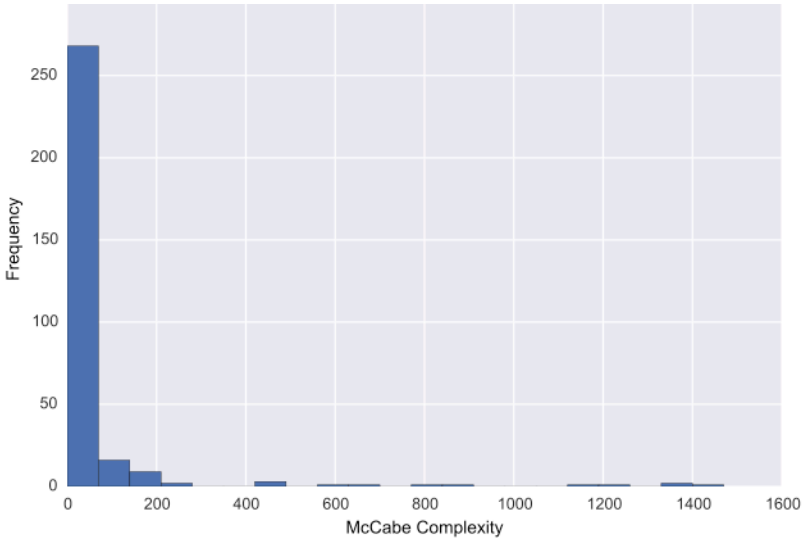


Figure 3-14: Distribution of cyclomatic complexity for laws passed by the 111th Congress

highly ranked with respect to all measures of complexity except coupling. For instance, it has the highest cyclomatic complexity, which is unsurprising since the Omnibus Appropriations Act contains multiple miscellaneous funding authorizations that should not permanently affect other areas of the U.S. Code of Law.[181] In contrast, a law such as PPACA is not only complex with respect to length and cyclomatic complexity, but also has a high degree of coupling with the rest of the U.S. Code, and has a large intersection with the largest strongly connected component (the core) of the U.S. Code.[182] In this respect, we can say that the Patient Protection Act has a higher impact on the U.S. Code than the Omnibus Appropriations Act.

We can explore this argument further by analyzing the network structure of these laws. Each piece of legislation affects a subset of the U.S. Code. While the overall U.S. Code is too large to visualize easily, the subsets of the U.S. Code modified by individual bills are small enough that visualization is helpful. As examples, Figure 3-15 shows PPACA, while Figure 3-16 shows the Omnibus Appropriations Bill from the 111th Congress.

[181] Omnibus Appropriations Act, 2009, Pub. L. No. 111-8, 123 Stat. 524.

[182] *See* Patient Protection and Affordable Care Act, Pub. L. No. 111-148, 124 Stat. 119 (2010). The U.S. Code's core contains the most interconnected sections. *See infra* App. D. Appendix A formally defines "core." *See infra* App. A.

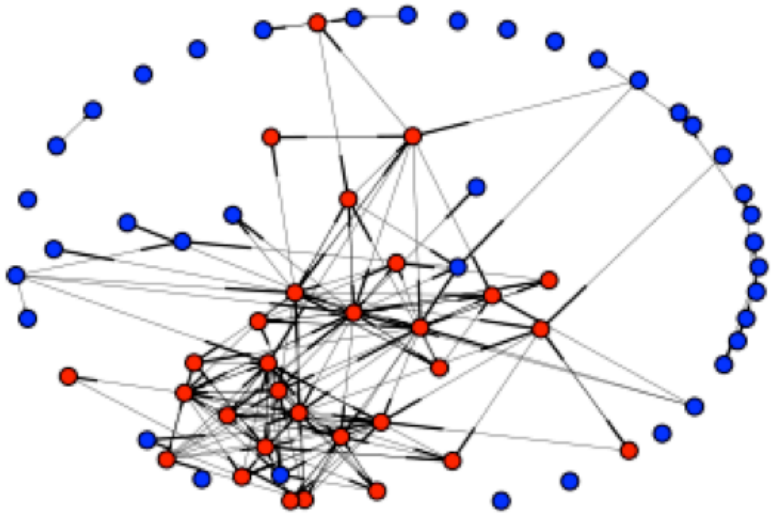


Figure 3-15: Sections of the U.S. Code modified by PPACA. Nodes in red belong to the largest connected component in this graph, which can be interpreted as the core of PPACA.

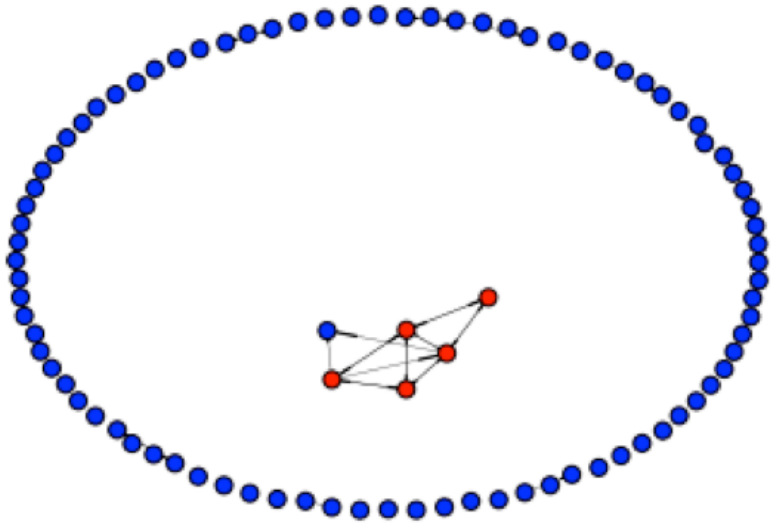


Figure 3-16: Sections of the U.S. Code modified by the Omnibus Appropriations Act of 2009. Nodes in red belong to the largest connected component in this graph, which can be interpreted as the core of the bill.

We highlight nodes in these networks with two colors. Nodes in red represent the *core of the law* ,[183] while nodes in blue represent the remaining sections of the law. That is, the nodes in red in Figure 3-15 represent the largest connected component of PPACA, while the nodes in red in Figure 3-16 represent the largest connected component of the Omnibus Appropriations Bill. Our graph layout algorithm places nodes in a circular fashion, with nodes with high levels of connectivity drawn more toward the center of the graph. As the figures indicate, the PPACA has many more interconnections between its sections. On the other hand, there are almost no cross-citations behind sections of the Omnibus Appropriations Bill. As Figure 3-16 shows, there are six sections that cite each other and form the core of the bill.[184] Thus, the Omnibus Appropriations Bill has a much lower degree of coupling than PPACA.

Appendices B and C show that these properties are not a fluke. Appendix B examines all appropriations bills passed since the 104th Congress. Each law corresponds to a figure in the appendix, which shows only the *core of the law* . As Appendix B shows, appropriations bills generally have very small cores. Appendix C, in contrast, shows the bills passed since the 104th Congress that have cores larger than 50.

One important conclusion from these results is that, even though appropriations bills are large, an expert reader can understand one section of it without needing to understand many other sections. In this respect, appropriations bills are simple. On the other hand, many of the sections of PPACA are coupled with other sections. To understand the impact of one section, an expert needs to understand the law as a whole, and may need to follow many levels of citations in the act. In this respect, the PPACA requires nonlinear, careful reading, making it very complex and challenging to understand. This insight emerges from examining the network structure of the law.

[183] *See infra* Appendix A
[184] *See* Omnibus Appropriations Act, 2009; *supra* Figure 3-16.

3.8 Structure of the Current U.S. Code: Titles 12 (Banks and Banking) and 26 (Internal Revenue Service)

In this section, we use our techniques to perform case studies of two very complex U.S. Code titles: Title 12 (Banks and Banking) and Title 26 (Internal Revenue Code).[185] Using our techniques, we can identify the sections with:

1. the highest complexity, according to the cyclomatic measure of conditional statement counts;

2. the highest degree of coupling, according to our core-periphery analysis.

Cyclomatic complexity will give us sections that have a high level of branching, and are therefore difficult to interpret without considering multiple conditional scenarios. The PageRank metric will show sections that, when modified, have a large probability of affecting other sections in their respective titles.[186]

3.8.1 Case Study of Title 12

Title 12 contains laws related to banks and banking institutions.[187] The banking sector in the United States has, as a result of consolidation and innovation, become more complex—today's financial institutions are involved in a wide array of transactions and activities that simply did not exist a generation ago.[188]

Along with multiple waves of financial crises and regulatory activity throughout the 20th and 21st centuries,[189] we argue that Title 12 can be challenging for the non-specialist to understand.[190] Our goal is to analyze and visualize the structure of

[185] *See* 12 U.S.C (2012); 26 U.S.C. (2012).
[186] *See supra* notes 125–26 and accompanying text; discussion *infra* Part VI.A.
[187] Banks and Banking, 12 U.S.C. (2012).
[188] *See* Lisa M. DeFerrari & David E. Palmer, *Supervision of Large Complex Banking Organizations* Fed. Res. Bull., Feb. 2001, at 47.
[189] *See* Laureen Snider, *The Conundrum of Financial Regulation: Origins, Controversies, and Prospects* , 7 Ann. Rev. L. & Soc. Sci. 121, 123–28 (2011).
[190] *See* 12 U.S.C (2012).

Table 3.8: Sections of Title 12 with Highest Cyclomatic Complexity

Section Number	Name	Number of Conditional Terms
§5390	Power and duties of the corporation	187
§1821	Insurance Funds	183
§1464	Federal savings associations	138
§1715l	Housing for moderate income and displaced families	130
§1467a	Regulation of holding companies	128

Title 12, as well as to pinpoint areas that are especially complicated. We do this by computing the cyclomatic complexity of each section of Title 12, and Table 3.8 reports the sections with the highest complexity. Any effort to reform banking regulation should begin with a systematic refactoring and simplification of these sections.

Another tool we can use is the network of citations that is produced by Title 12. This network can be visualized in Figure 3-17 , which shows a very dense graph. Nodes highlighted in red correspond to the core of this graph, and make up a significant fraction of Title 12. Thus, even a slight modification to a section of Title 12 is likely to have large repercussions across all other sections, and Figure 3-17 provides a systematic way to gauge such repercussions before any modification is implemented.

While it is helpful to visualize Title 12 as a network in this way, it is hard to specify the most "influential" sections of the Title just by looking at this network. It would seem from the metrics so far that all sections in the core of Title 12 would be just as influential. In order to break this tie, we introduce the PageRank metric, which is frequently used in network analysis and has been used to rank the importance of web pages for Internet search engines.

Table 3.9 gives the nodes in Title 12 with the highest PageRank. The one with the highest PageRank (and therefore the most influential under this metric) is 12 U.S.C. § 1481, which is the Bank Holding Company Act's definitions section.[191] This

[191] 12 U.S.C. 1481 (2012).

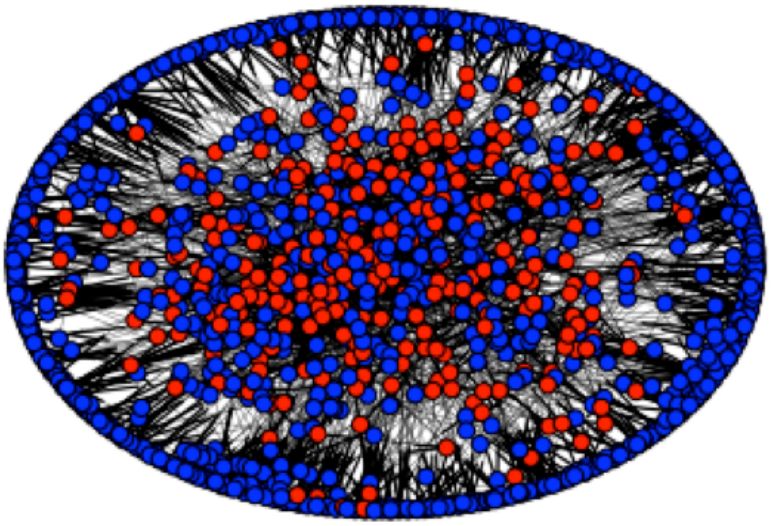


Figure 3-17: Core-Periphery Network of Title 12 (Banks and Banking)

fact suggests that, if the definitions in this section were to be amended by a financial reform, then it would have a significant impact on the interpretation of all other sections of Title 12.

3.8.2 Case Study of Title 26 (Internal Revenue Code)

We apply the same analysis to Title 26 (Internal Revenue Code), which is known to be a very complex title of the U.S. Code.[192] Figure 3-18 shows the network structure induced by cross-citations in Title 26, with nodes in red again showing nodes that are in the core of the title. As we can see, Title 26 is even denser and has a larger core than Title 12.

We can use cyclomatic complexity and PageRank to find significantly complex sections of Title 26. Table 3.10 shows the Title 26 sections with the highest cyclomatic complexity. Table 3.11 gives the sections in Title 26 with the highest PageRank. The one with the highest PageRank (and therefore the most influential under this metric) is 26 U.S.C. §501, which defines exemptions from taxation. As in our analysis of Title

[192]The 2012 National Taxpayer Advocate's Report to Congress declared: "The most serious problem facing taxpayers—and the IRS—is the complexity of the Internal Revenue Code (tax code)." Nat'l Taxpayer Advocate, 2012 Annual Report to Congress 3 (2012),`http://www.taxpayeradvocate.irs.gov/2012-Annual-Report/downloads/Most-Serious-Problems-Tax-Code-Complexity.pdf` . The Report estimated that Americans spend 6.1 billion hours per year on tax compliance. *Id.*

Table 3.9: Title 12 Sections with Highest PageRank

Section Number	Name	Beginning Excerpt
1841	Bank Holding Company Act Definitions	Except as provided in paragraph (5) of this subsection, "bank holding company" means any company which has control over any bank or over any company that is or becomes a bank holding company by virtue of this chapter.
101	Repealed	Section 101, acts Mar. 14, 1900, ch. 41, §12, 31 Stat. 49; Oct. 5, 1917, ch. 74, §2,40 Stat. 342, provided for delivery of circulating notes in blank to national banking associations depositing bonds with Treasurer of United States.
1818	Termination of Status as Insured Depository Institution	(a) Termination of insurance Voluntary termination Any insured depository institution which is not (A) a national member bank; (B) a State member bank; (C) a Federal branch; (D) a Federal savings association; or (E) an insured branch which is required to be insured under subsection (a) or (b) of section 3104 of this title, may terminate such depository institution's status as an insured depository institution if such insured institution provides written notice to the Corporation of the institution's intent to terminate such status not less than 90 days before the effective date of such termination.
1709	Insurance of Mortgages	(a) Authorization The Secretary is authorized, upon application by the mortgagee, to insure as hereinafter provided any mortgage offered to him which is eligible for insurance as hereinafter provided, and, upon such terms as the Secretary may prescribe, to make commitments for the insuring of such mortgages prior to the date of their execution or disbursement thereon.
1813	Federal Deposit Insurance Act Definitions	(a) Definitions of bank and related terms (1) Bank The term "bank" — (A) means any national bank and State bank, and any Federal branch and insured branch; (B) includes any former savings association.

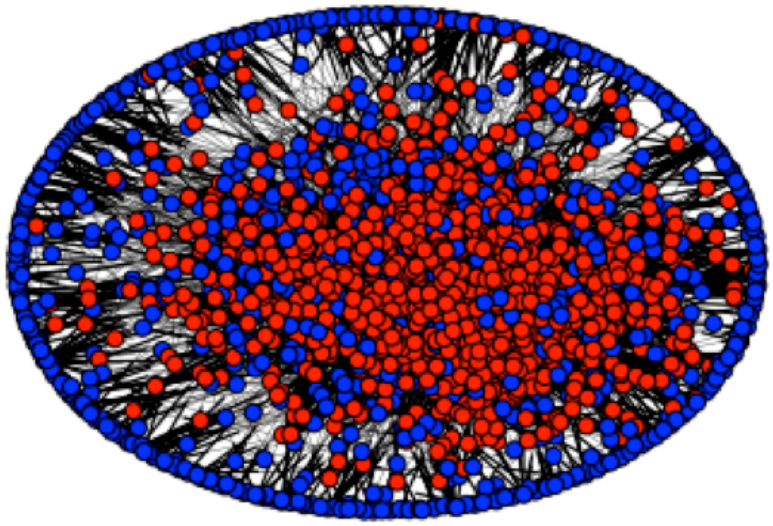


Figure 3-18: Core-Periphery Network of Title 26 (Internal Revenue Service)

Table 3.10: Sections of Title 26 with Highest Cyclomatic Complexity

Section Number	Name	Number of Conditional Terms
§168	Accelerated cost recovery system	392
§401	Qualified pension, profit-sharing, and stock bonus plan	344
§141	Private activity bond; qualified bond	213
§3121	Definitions [Subchapter C - General Provisions]	201
§42	Low-income housing credit	195

12, this fact implies that changing these exemptions would have a wide reaching effect on the rest of Title 26.

3.8.3 Comparing Titles 12 and 26 to Other Titles

As seen in our visualizations both Title 12 and Title 26 have very large cores, implying great complexity. A natural question is whether this characteristic is common to all titles of the U.S. Code. As elaborated more fully in Appendix D, this is not the case. In fact, Titles 12 and 26 have two of the largest cores in the U.S. Code. The top 5

Table 3.11: Title 26 Sections with Highest PageRank

Section	Name	First Clause
501	Exemption from tax on corporations, certain trusts, etc.	(a) Exemption from taxation An organization described in subsection (c) or (d) or section 401 (a) shall be exempt from taxation under this subtitle unless such exemption is denied under section 502 or 503.
1564	Repealed	Section, added Pub. L. 91-172, title IV, §401(b)(1), Dec. 30, 1969, 83 Stat. 600; amended Pub. L. 94-455, title XIX, §§1901(b)(1)(J)(vi), (21)(A)(ii), 1906(b)(13)(A), Oct. 4, 1976, 90 Stat. 1791, 1797, 1834, related to transitional rules in the case of certain controlled corporations.
1	Tax Imposed	(a) Married individuals filing joint returns and surviving spouses There is hereby imposed on the taxable income of — (1) every married individual (as defined in section 7703) who makes a single return jointly with his spouse under section 6013, and (2) every surviving spouse (as defined in section 2 (a)), a tax determined in accordance with the following table: If taxable income is: / The tax is: Not over $36,900 : 15% of taxable income. Over $36,900 but not over $89,150 : $5,535, plus 28% of the excess over $36,900. Over $89,150 but not over $140,000 : $20,165, plus 31% of the excess over $89,150. Over $140,000 but not over $250,000 : $35,928.50, plus 36% of the excess over $140,000. Over $250,000 : $75,528.50, plus 39.6% of the excess over $250,000.
170	Charitable, etc., contributions and gifts	(a) Allowance of deduction (1) General rule There shall be allowed as a deduction any charitable contribution (as defined in subsection (c)) payment of which is made within the taxable year. A charitable contribution shall be allowable as a deduction only if verified under regulations prescribed by the Secretary.
401	Qualified pension, profit-sharing and stock bonus plans	(a) Requirements for qualification A trust created or organized in the United States and forming part of a stock bonus, pension, or profit-sharing plan of an employer for the exclusive benefit of his employees or their beneficiaries shall constitute a qualified trust under this section — (1) if contributions are made to the trust by such employer, or employees, or both, or by another employer who is entitled to deduct his contributions under section 404 (a)(3)(B) (relating to deduction for contributions to profit-sharing and stock bonus plans), or by a charitable remainder trust pursuant to a qualified gratuitous transfer (as defined in section 664 (g)(1)), for the purpose of distributing to such employees or their beneficiaries the corpus and income of the fund accumulated by the trust in accordance with such plan...

Table 3.12: U.S. Code Titles with Largest Cores

Title	Core Size (Number of Sections)
26 (Internal Revenue Code)	1037
42 (Public Health and Welfare)	873
12 (Banks and Banking)	279
20 (Education)	234
49 (Transportation)	200

titles with the largest cores are given in Table 3.12 . The average core size of a U.S. Code title is 89.81, much lower than the size of the cores of Titles 12 and 26. In Appendix D, we show visualizations of the cores of all U.S. Code titles, illustrating how rare it is to have a very large and dense core. Thus, our techniques seem to be useful indicators of complexity for a given title.

3.9 Conclusion

The similarities between software and law is striking—in many respects, law *is* code. When viewed from a software engineering perspective, the U.S. Code resembles a large software system, and the application of software design principles allows us to quantify the extent to which the law is concise, changing, coupled, and complex. Our methods reveal the rise, spread, and fall of legal terms used in the U.S. Code, the structure of the cross-references network, and the types of laws that Congress enacts. When applied to specific titles, these methods have identified particularly complex and highly interconnected sections, which should be prime candidates for regulatory reform and simplification. The sheer size and number of cross references within the core sections imply that software-engineering methods can play an important role in leveraging human ability. Therefore, a software engineering approach to measuring and managing the U.S. Code allows lawmakers to enact better legislation with fewer vulnerabilities.

Creating less complex laws and simplifying the existing legal code also reduce the

number of unintended consequences and ensure more fair and equitable outcomes for all stakeholders. By developing a more coherent and systematic view of the entirety of the body of laws governing our society, we create more informed participants in the legal system, empowering lawyers, judges, and individual citizens in their respective roles of proposing, enforcing, interpreting, and changing the law. One cannot manage what one does not measure, and as the U.S. Code becomes larger and more unwieldy, software-engineering methods can greatly enhance our ability to participate in the legislative process.

Chapter 4

Text Reuse and Financial Crisis Policy Trajectories in Congress

Preamble

This chapter includes content adapted from the following paper:

Li, W., Larochelle, D., Lo, A. W. Estimating Policy Trajectories During the Financial Crisis. NLP Unshared Task in PoliInformatics, June 2014 [28].

4.1 Introduction: "Legitimate" Text Reuse in Legal and Political Texts

Thus far, this thesis has explored features of legal and political texts at multiple levels: In Chapter 2, common n-grams (up to three words in length) are shown to be predictive of Supreme Court authorship; in Chapter 3, meanwhile, document characteristics such as length, cross-references, and conditional statements in the United States Code provide insights into the structure and evolution of the United States Code.

A common attribute of both of these previous chapters is that they both use observations of repeated text (albeit of different lengths) to find interesting patterns

and insights. The remaining two chapters of this thesis explore text reuse in further detail: This chapter focuses on text reuse that occurs in key Congressional bills during the Financial Crisis, while Chapter 5 introduces a novel probabilistic model for text reuse and an application to free-form public comments on federal regulations.

In some contexts, text reuse occurs in the form of plagiarism, i.e. misappropriating someone else's ideas as one's own. In legal and political texts, though, such re-occurring or copied text can occur legitimately in a number of situations:

1. **Common legal language:** Drafters of new documents may base their writing on existing documents, or passages may have been approved by lawyers or other authorities. As a result, large sections of text may occur even in entirely different bills, contracts, or other documents. Depending on context, this repeated text may be considered "boilerplate" or highly material to the subject of the document.

2. **Multiple versions of historical documents:** Over time, documents may be revised, meaning that two different versions may have large amounts of unchanged, similar text. The structure of text reuse in these contexts, such as in titles of the U.S. Code from different years in Chapter 3, can reveal the nature and extent of changes. This chapter explores text reuse within a single two-year session of Congress, a much shorter period of time.

3. **Text reuse as speech:** Chapter 5 introduces a novel probabilistic model for text reuse that, among other applications, can be used to quantify the occurrence of ideas in a large corpus of text. Measuring the size and viewpoints of large groups of people who state the same idea identically or similarly is the focus of the next chapter.

Text reuse has also been the subject of growing academic interest in computational social science, particularly on historical and political corpora. For example, the spread of political speech in 19th-century American newspapers, which often featured reprints of speeches and other articles, has been the subject of study by

historians and computer scientists [49]. Other recent work has examined different mechanisms of text reuse, relating them directly to types of observed overlapping n-grams: short n-gram overlaps correspond to *shared subject matter*, moderate n-gram overlaps are a result of *shared rhetorical goals*, and longer n-gram overlaps are due to *shared sources* [30]. Meanwhile, the work described in this chapter largely stems from the study of text reuse in different iterations of bills in legislative bodies, particularly in Congress [49, 56]. Other work in this domain has focused on the influence of lobbyists and template bills on legislation, particularly at the state level [20, 21]. In addition, beyond academic research, text reuse has begun to make its way into applications by civil society groups: projects such as Churnalism by the UK Media Standards Trust [36] and the US Sunlight Foundation seek to find reused text from corporate press releases in news articles, while groups like Data Science for Social Good have employed such methods for legislative bills [10].

This expanding body of applications of text reuse suggests that it is a powerful approach to understanding public speech, and also motivates the development of novel models that describe text reuse, such as the one described in Chapter 5.

This chapter focuses on text reuse as a historical marker of political document evolution: We examine the trajectories of policy ideas that eventually become law. Specifically, we examine the trajectories of ideas contained in four bills related to the Financial Crisis during the 110th (2007-08) and 111th (2009-10) Congresses. By identifying the first appearance of bill text, visualizing the results, and constructing metrics to quantify the congressional "consideration time" of a bill's ideas, our analysis reveals that two of the four bills were dominated by ideas that were first introduced many months before their eventual passage, while the other two bills contained mostly new text and were truly novel responses to the Crisis. In addition, we also apply the method to find policy ideas related to the Financial Crisis that were not included in successful bills. In Chapter 5, we turn our focus to finding text reuse in public speech, thereby illustrating the wide utility of models of text reuse in legal and political texts.

4.2 Text Reuse in Financial Crisis Legislation

The Financial Crisis of 2007–2009 had serious impacts on housing markets, the banking system, the economy, and society as a whole, compelling the U.S. government to intervene. In this chapter, we ask questions about the federal policy response: what policy ideas were invented in response to major events during the crisis, and which ideas had been under consideration for longer periods of time?

We focus on the United States Congress's legislative activity between 2007 and 2010. Specifically, we apply text reuse methods to trace the trajectories of policy ideas contained in four key bills related to the Financial Crisis. Our work, as described further below, reveals interesting timeline-based patterns of congressional lawmaking activity for these bills.

4.3 Related Work

Our analysis applies the text-reuse approach to analyzing congressional bills introduced by Wilkerson et al. [56]; we focus on bills related to the Financial Crisis and compares the patterns of legislative activity that emerge from this analysis. In this work, we adopt a simpler approach to finding similar bill sections: We do not perform local sequence alignment on closely matching sections, and instead use length and Jaccard coefficient features to classify sections as "matched" and "not matched>." We also propose metrics that quantify a bill's congressional "consideration time" and extend its application to unsuccessful policy ideas.

4.4 Dataset and Methodology

For this work, we obtained the text of all bills introduced in the 110th and 111th Congresses from the Congressional Bills Project [4]. The dataset includes bills partitioned by section, along with the bill's date of issue, chamber of introduction (House or Senate), and version. Table 1 lists the number of bills introduced and enrolled in these two congresses.

Table 4.1: Summary of Bills in 110th and 111th Congresses

Congress	# of Bills Introduced in House	# of Bills Introduced in Senate	Total Introduced	Total Enrolled
110th (2007-08)	10,437	4,755	15,192	439
111th (2009-10)	8,923	4,795	13,718	367

The following four bills related to the Financial Crisis were analyzed:

- **Housing and Economic Recovery Act (HERA) of 2008**: A law designed to address the subprime mortgage crisis.
- **Emergency Economic Stabilization Act of 2008**: This law included the Troubled Asset Relief Program (TARP), a response to the credit crunch and the risk of failing banks.
- **American Recovery and Reinvestment Act (ARRA) of 2009**: A stimulus bill that made infrastructure investments and aimed to create jobs.
- **Dodd-Frank Wall Street Reform and Consumer Financial Protection Act of 2010**: A reform bill with substantial changes to American financial regulations.

Our goal is to find instances of the text of these four final, enacted laws in earlier bills in the 110th and 111th Congresses. Specifically, given that we have a corpus of bill *sections*, we seek to find similar sections from earlier bills.

4.4.1 Finding Similar Sections

To find similar bill sections, we first need a definition of similarity between bill sections. We adopt an approach in which two sections that share a large number of common words are considered similar; specifically, we use the Jaccard coefficient, a widely used similarity metric in natural language processing and other domains [26].

For sets of words A and B, the Jaccard coefficient varies between 0.0 (if A and B have no shared words) and 1.0 (for $A = B$) and is defined as follows:

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{4.1}$$

Other measures of similarity include the Sørensen-Dice coefficient, which provides a normalized similarity score and has been used in other text reuse applications [e.g. 49, 56]. These applications also often align two passages of text using local or global sequence alignment algorithms in order to precisely identify the regions of similarity within a section. For the purpose of this work, however, we simply use the Jaccard coefficient as a feature for ease of implementation and because it can be computed more quickly than the optimal local sequence alignment. More precisely, given two sequences of words A and B, with N_A and N_B numbers of words, respectively, it can be shown that the Jaccard coefficient can be computed in $O(N_A + N_B)$ operations, while sequence alignment requires $O(N_A \cdot N_B)$ operations.

In addition to the time required to compute the similarity metric between two passages, a second challenge of text reuse systems is computing the similarity between a query passage and all of the passages in the database. In the case of the Congressional Bills dataset of this chapter, there are more than 250,000 bill sections; the naive, pairwise approach to computing the Jaccard coefficient between even a single query bill section and all bill sections is impractical; for a dataset of D documents, the time complexity would be $O(N \cdot D)$. To overcome this issue, we built an inverted index (hash table) that maps words to bill sections for every bill introduced in the 110th and 111th Congresses. Instead of $O(N \cdot D)$ operations, finding the sets of documents for each of the N query words requires just $O(N)$ lookups, and sorting the resulting short list of documents by the number of words that they contain can also be done quickly. For the four bills of interest in this chapter, we used the following procedure:

- For each bill section (the "target section"), the 100 bill sections (out of 259,418) with the most matching words were determined using this hash table method.

These top 100 sections are the "candidate matched sections."

- For each "candidate matched section" and the target section, we computed the Jaccard coefficient, which produces a better re-ranking of "candidate matched sections" than simply the number of matching words.
- As described further below, the length, Jaccard coefficient, and other decision rules were used to determine whether sections were true matches. The true matches correspond to earlier instances of the text that was part of the final enrolled bill.

Visualizations of this procedure for the four bills of interest are shown in the next section.

4.4.2 Classifying Matched Sections

Figure 4-1 shows the distribution of target section word lengths and Jaccard coefficients for 245 bill sections paired with query sections from the Emergency Economic Stabilization Act of 2008, which includes TARP. We examined each pair of sections manually labeled them as "matched" or "non-matched". In general, shorter sections required a higher Jaccard coefficient in order to be classified as a true match; for example, the "Short Title" sections of entirely unrelated bills often contained almost identical language, except for the short title itself. The length and Jaccard coefficient are informative features that help separate matches from non-matches.

For our purposes, we also excluded certain "boilerplate" sections that occur in many bills, such as severability clauses. We used this ground-truth labeling for TARP. For the other three bills, based on Figure 4-1, we used a Jaccard coefficient threshold of 0.88 for sections less than 500 words long and a threshold of 0.75 for longer sections.

4.5 Results and Visualization

Figures 4-2, 4-3, 4-4, and 4-5 show when the sections of the four target bills were introduced in the 110th or 111th Congresses. Repeated runs of contiguous sections

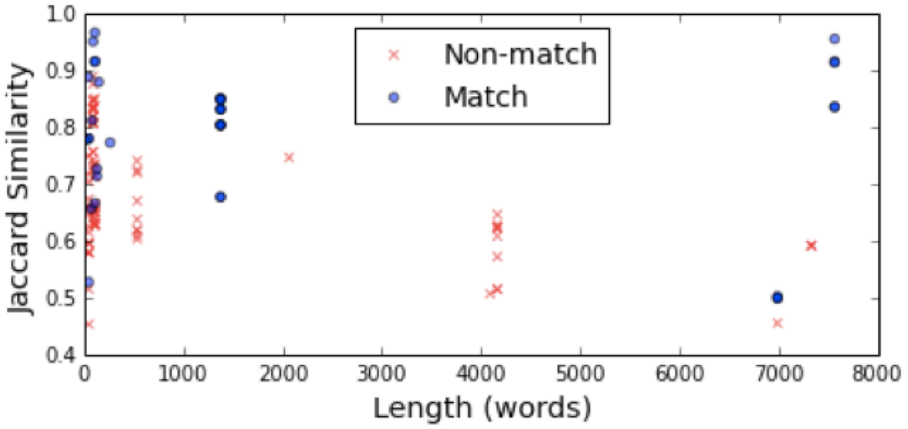


Figure 4-1: Jaccard coefficients and lengths for 246 labeled matching and non-matching bill sections.

are annotated with their policy content. These plots reveal interesting insights about the dynamics of legislation in Congress:

4.5.1 Housing and Economic Recovery Act (HERA) of 2008

Figure 4-2 illustrates the trajectory of various policy ideas that were incorporated into the Housing and Economic Reform Act (HERA) of 2008. Its provisions included the creation of the Federal Housing Finance Industry, which was empowered to place the government-sponsored enterprises Fannie Mae and Freddie Mac under government conservatorship (it would do so in September 2008 [23].

We label five regions with varying length and consideration times that correspond to different kinds of policies. This figure underscores how bills can contain a wide range of policy ideas; in fact, at least three "Acts" (the Federal Housing Finance Regulatory Reform Act, the Secure and Fair Enforcement of Mortgage Licensing Act, and the FHA Modernization Act) are all part of this bill. Rising economic uncertainties and the growing housing crisis led to the passage of all of these pieces of legislation as part of HERA in July 2008.

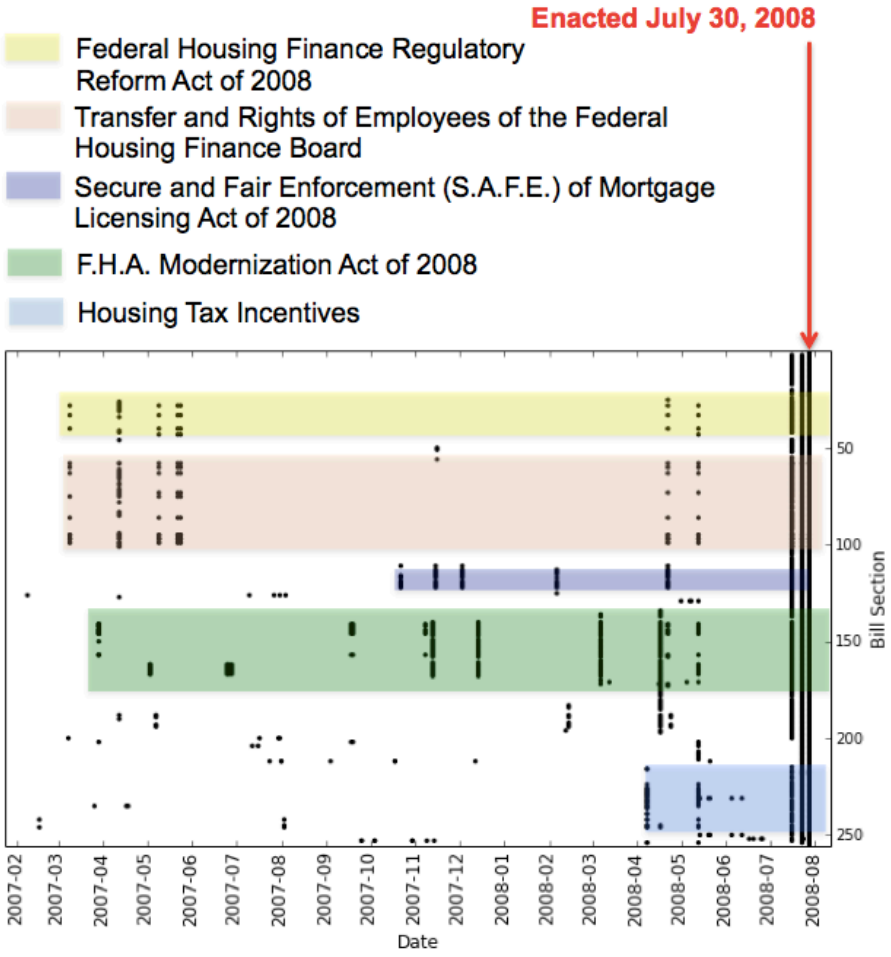


Figure 4-2: Housing and Economic Recovery Act (HERA)

4.5.2 Emergency Economic Stabilization Act of 2008 (including TARP)

In Figure 4-3, we annotate the contents of the Emergency Economic Stabilization Act, which includes, as the first part of the bill, the Troubled Asset Relief Program (TARP) that created a $700 billion fund to purchase assets and equity from failing financial institutions [39]. . Given the politically unpalatable nature of such a bailout, the U.S. House of Representatives voted against the bill on September 29, 2008, causing the Dow Jones Industrial Average to fall 7 percent, or 777 points, which, as of the publication of this thesis, still remains the largest one-day drop in the history of the stock market index.

Figure 4-3 is notable for what it reveals about the contents of the Emergency

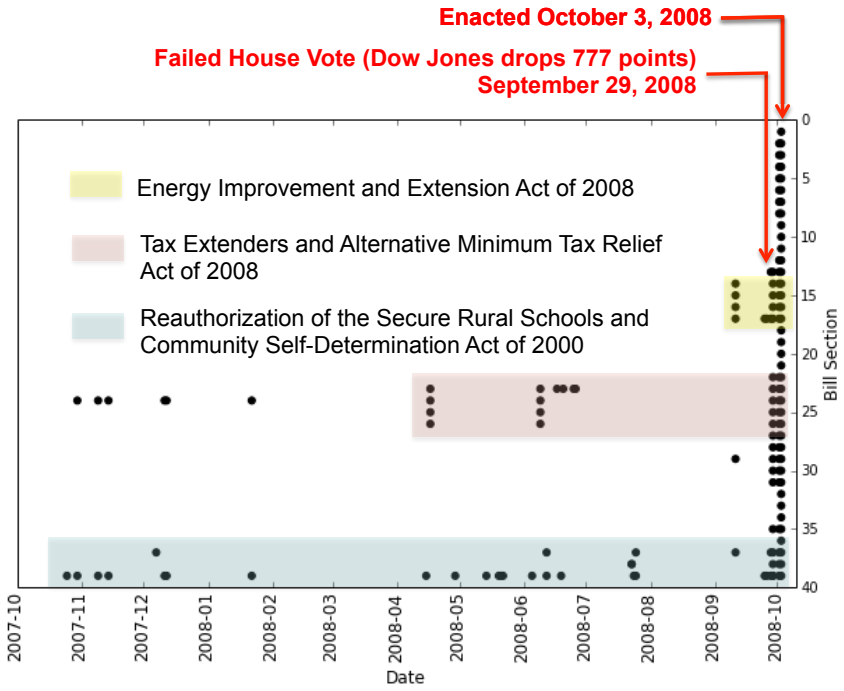


Figure 4-3: Troubled Asset Relief Program (TARP)

Economic Stabilization Act: It contains legislation related to renewable energy tax credits, a stalled tax bill that was a point of contention between Democrats and Republicans, and provisions related to Secure Rural Schools Act, which relates to emergency services in rural communities. These policy riders, which, at best, have a tenuous relationship to TARP, were inserted into the successful bill to obtain the majorities needed for it to pass the House and Senate. This figure demonstrates how bills are merely vehicles for policies, and the dealmaking that goes into the lawmaking process: Any bill that can assemble a majority of legislators to vote for it can become enacted law. Finding these coalitions of legislators and viable bill vehicles can be a substantial part of the role of any Member of Congress and their staff.

4.5.3 American Recovery and Reinvestment Act (ARRA) of 2009

Figure 4-4 shows the trajectories of policy ideas of President Barack Obama's early-first-term economic stimulus package, enacted in February 2009. In contrast to the

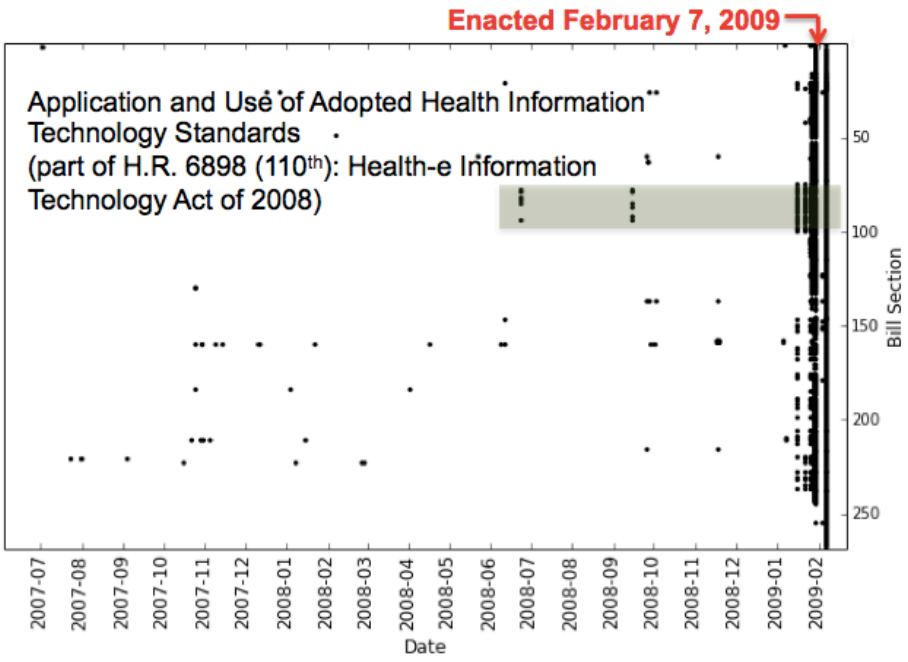


Figure 4-4: American Recovery and Reinvestment Act

three other bills, this enacted law contains relatively few long-considered policy ideas, with the only exception being a set of health information technology provisions from the previous Congress. As a signature bill of a new presidential administration, it is perhaps unsurprising that ARRA contains a relatively large amount of original content, with little reused text from the preceding two-year period.

4.5.4 Dodd-Frank Wall Street Reform and Consumer Protection Act of 2010

The Dodd-Frank law, shown in Figure 4-5, was a major piece of financial reform legislation with sweeping impacts on many different aspects of financial regulations. In this plot, the varying consideration times for different sections of Dodd-Frank are interesting. While the analysis of "regulatory buildup" is beyond the scope of this chapter, Figure 4-5 suggests how text reuse might be used to study the existence and extent of such a phenomenon.

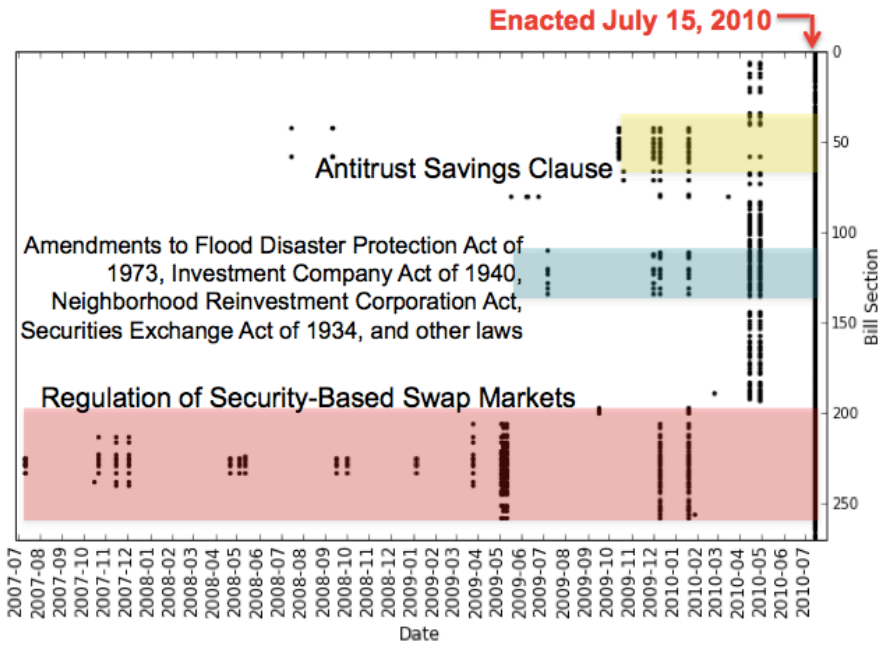


Figure 4-5: Dodd-Frank Wall Street Reform and Consumer Protection Act

4.6 Bill Consideration Time Metrics

The visualization of these four bills illustrates how many sections were actually introduced in Congress days, months, or even years earlier. We can define a summary metric of the "consideration time" metric in two ways:

- Week Threshold (WT): The percentage of bill sections introduced more than k weeks prior to the passage of the bill. Based on these four financial bills, we set $k = 12$, corresponding to about three months.

- Average gestation (AG): The average time that the sections of a bill were considered. Formally, if a bill has n sections, had its final successful vote on day D_p, and section i was introduced on day D_i:

$$AG = \frac{1}{n}\sum_{i=1}^{n}(D_p - D_i) \tag{4.2}$$

The values of these two metrics for each of the four bills under study are shown in Table 2. They capture how HERA and Dodd-Frank contain provisions that had long

Table 4.2: Consideration Time Metrics for Financial Crisis Bills

Bill	WT (k=12)	AG (weeks)
HERA	0.541	26.0
TARP	0.154	7.2
ARRA	0.086	5.8
Dodd-Frank	0.551	21.4

consideration times in Congress, while TARP (as an urgent response to the collapse of systemically important financial institutions) and ARRA (as the product of the new Obama administration) were considered for much shorter periods.

4.7 Analysis and Discussion

4.7.1 Consideration Times of Financial Crisis Bills

The results in Table 4.2 show that TARP and ARRA had relatively short consideration times compared to HERA and Dodd-Frank. TARP was a response to the failure of Lehman Brothers on September 14, 2008—time was of the essence, compelling Congress to quickly draft legislation enabling the federal government to buy toxic assets from financial institutions. Such a plan, it appears, simply did not exist prior to September 2008, or at least was not considered on the floor of the House or Senate. Meanwhile, ARRA's consideration time metrics are also short, albeit for different reasons: it was the product of a new presidential administration, and other than one part focused on electronic health records, it appears to have contained new policy ideas.

The sections added to these two bills were, arguably, peripheral to their respective core missions. These "policy riders" are typically discussed in the context of appropriations bills and are the subject of substantial controversy []. The political urgency associated with TARP and ARRA seem to have made them suitable for policy rider inclusion.

In contrast, HERA and Dodd-Frank had longer gestation times. Their long-considered bill sections appear to be germane to the overall goal of the bill. One

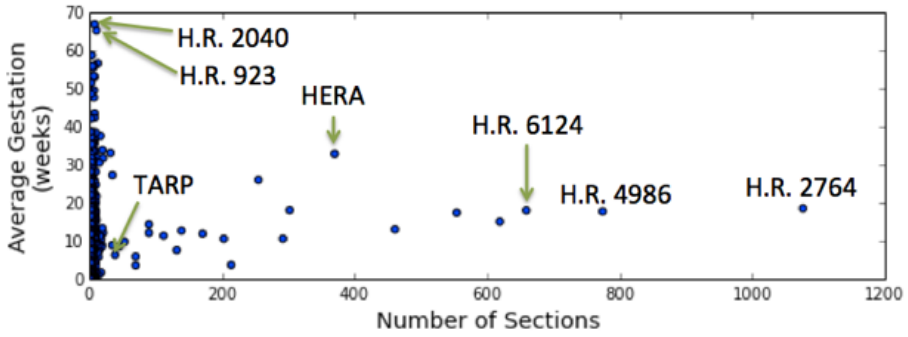


Figure 4-6: Bill sizes and average gestation times in 110th Congress.

extension of this work would be to identify whether these ideas were actually introduced in earlier Congresses or in other kinds of political documents.

4.7.2 Summarizing Congressional Lawmaking Activity

We have focused on the Financial Crisis through the lens of lawmaking activity. We chose to explore bill text for the following reasons:

- Congressional bills are the distillation of the debates surrounding policy ideas – they contain text that could potentially become law.
- The visualization of enrolled bills is inherently biased toward successful ideas, which inevitably discounts other dimensions of the data, such as the structure of previously introduced bills. However, focusing on policy ideas with high "fitness" seems both reasonable and useful.

4.7.3 Distribution of Consideration Times

For comparison, Figure 4-6 shows the bill sizes and average gestation times of 439 enrolled bills in the 110th Congress, for which the dataset had a complete set of dates for enrolled bills.

Somewhat surprisingly, the 110th Congress did not appear to consider bills with the most number of sections for longer periods of time: the average gestation seems insensitive to the number of sections. Similarly, there did not seem to be a relationship between the week threshold metric and bill size.

Table 4.3: S. 2338 Policy Sections Excluded from HERA

Bill Section	Title	Maximum Jaccard Score
102	Maximum principal loan obligation	0.68
103	Cash investment requirement and prohibition of seller-funded downpayment assistance	0.82
112	Home equity conversion mortgages	0.57
123	Moratorium on implementation of risk-based premiums	0.49
210	Leasehold requirements	0.81

Along with TARP and HERA, we annotated Figure 4-6 with the following bills:

- H.R. 2764, the bill with the most number of sections in the 110th Congress, is the Consolidated Appropriations Act of 2008.

- H.R. 4986 and H.R. 6124 are the National Defense Authorization Act and the Food, Conservation, and Energy Act, respectively.

- The two bills with the longest AGs were H.R. 2040 (the Civil Rights Act of 1964 Commemorative Coin Act) and H.R. 923 (the Emmett Till Unsolved Civil Rights Act of 2007), both sponsored by Rep. John Lewis (D-GA). Interestingly, H.R. 923 was delayed because Sen. Tom Coburn (R-OK) placed a hold on it until it passed in September 2008.

4.7.4 Finding Unsuccessful Policy Ideas

The method applied in this project can also be used to find Financial Crisis-related policies that did not become law. As a case study, we selected S. 2338 (the FHA Modernization Act of 2007) from the 110th Congress. The visualization (Figure 4-7) now includes markers *after* its passage in the Senate in December 2007. S. 2338 has twelve sections that are "non-matches" with HERA, seven of which are structural or boilerplate elements. Table 4.3 lists the titles of the other five excluded sections and their maximum Jaccard coefficient with any HERA section.

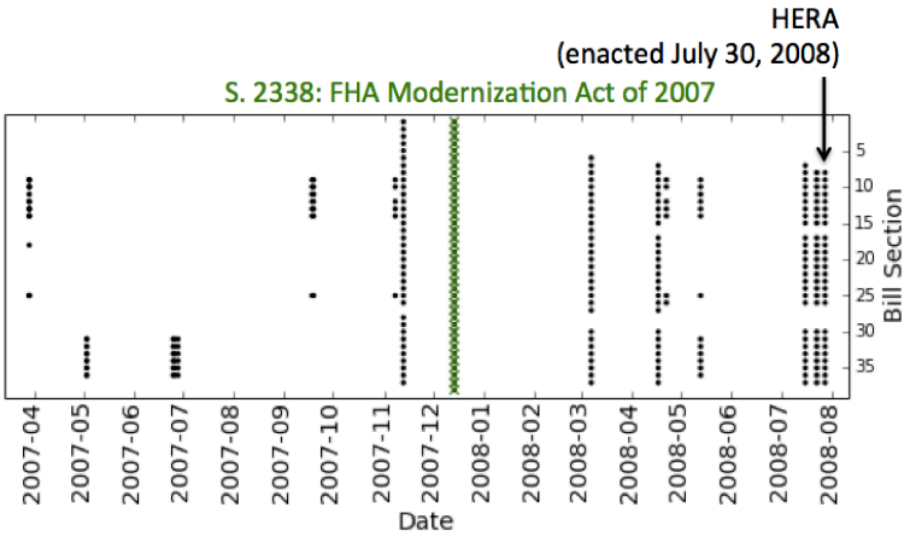


Figure 4-7: Trajectories of sections of S. 2338, FHA Modernization Act of 2007

For some of these sections, the maximum Jaccard coefficient suggests overlaps with parts of HERA. For example, HERA also includes a section entitled "Maximum principal loan obligation," but the HERA version has additional text related to the "treatment of up-front premiums." This clause's inclusion reduced the Jaccard coefficient of Section 102 of S. 2338 to below the threshold required to be a "matched" section.

4.8 Limitations and Further Work

Conceptually, parsing bills into sections is logical — as Wilkerson et al. [56] note, Section 104 of Title 1 of the United States Code states that a section "shall contain, as nearly as may be, a single proposition of enactment." In practice, bill sections differ substantially in length and policy content. As well, ambiguity may exist about the definition of a section—all 36 sections of Title 1 of the TARP part of the Emergency Economic Stabilization Act, for instance, were defined as a single "section" in the dataset. For consistency, we did not attempt to re-section the bills.

Meanwhile, compressing a bill section into a single dot is clearly a simplification. A worthwhile extension could be incorporating interactivity, such as zooming into specific time frames or clicking points to reveal the original bill text.

Beyond studying the Financial Crisis, this text-based approach could have value as an open-government tool. For example, it is already possible, through a web interface called "Scout," to receive notifications when a vote on a bill takes place in Congress [46]. Following the trajectory of highly similar passages of text through the lawmaking process could help citizens, journalists, and other interested parties efficiently keep track of the progress of policy ideas through Congress.

4.9 Conclusions

In this chapter, we provide an estimation of policy trajectories in four significant bills passed by Congress related to the Financial Crisis. The visualization makes it possible to determine which policy proposals were considered in Congress for longer periods of time, and which parts of these bills were genuinely new. Our "week threshold" (WT) and "average gestation" (AG) metrics reveal that HERA and Dodd-Frank had much longer consideration periods than TARP and ARRA. The computational approach presented here could be useful for promoting greater understanding and accountability of Congress.

Chapter 5

Probabilistic Text Reuse

Overview

Many collections of related documents contain passages of text that are highly similar. Discovering these passages can be an informative way to represent and understand interesting hidden structure in the document collection. In this paper, we present a new probabilistic model for document collections with repeated text passages. Our Probabilistic Text Reuse (PTR) model takes a generative approach and assumes that documents are composed of passages, where passages are either drawn from a canonical set of word sequences or *ideas*, or from a background language model. We model ideas as probabilistic finite state transducers, which generate each of the words in our sequence with some probability of matching, substitution, addition, or deletion. We also present algorithm for learning these ideas and their locations. Finally, we illustrate the utility of our model by finding common ideas in a large set of public comments on proposed U.S. net neutrality regulations and repeated sections of text in U.S. congressional bills. This paper provides a global objective function for text reuse, algorithms for optimizing this function, and an application for understanding a large collection of documents.

5.1 Introduction

Making sense of a large collection of text documents is an important task in many domains. For example, in the realm of politics and public policy, politicians may receive large numbers of letters from their constituents, and government agencies may receive millions of comments on proposed legislation. Citizens and journalists may be interested in the evolution of bills, regulations, and other documents released by government agencies or whistleblowers. Manually reading these large collections of documents may be impractical; automated and accurate ways of representing the content and viewpoints of these datasets are needed.

In each of the aforementioned tasks, the collections of text documents have a common characteristic: there are many instances of reused text. In some cases, such as when an advocacy organization encourages people to submit a form letter, entire documents can be repeated; in others, such as a collection of all bills introduced in a legislative body, smaller parts of documents may be reused. These duplicates can also be noisy, either due to technical processes (such as optical character recognition or text formatting variations) or human editing efforts (such as changes in wording or accidental copying mistakes or changes). Research in text reuse [e.g. 49, 56, 28] aims to understand documents through patterns of repeated or approximately repeated text.

In this chapter, we present a principled, probabilistic approach to capture and quantify these instances of text reuse in a corpus. We describe Probabilistic Text Reuse (PTR), a generative model for text reuse. Our PTR model explicitly learns a latent set of canonical text passages, which we term "ideas," that are directly useful to a person seeking to make sense of a large collection of documents. Our approach allows us to handle uncertainty and noise in text reuse — many documents may contain the same "idea" but with slight variations, and we can discover those examples in a principled manner. In this way, we take an important step towards automatically discovering common ideas from such large collections.

The contributions of our work are as follows:

1. We present Probabilistic Text Reuse (PTR), a generative model for making sense of collections of documents that have instances of repeated text. To the best of our knowledge, the PTR model is the first text reuse model with a global objective function, enabling patterns of text reuse to be evaluated in a principled, quantitative manner.

2. We present scalable algorithms for finding good parameter settings for PTR, including the discovery of ideas, partitions, and assignments.

3. We show the utility of PTR by illustrating how it is helpful for making sense of tens of thousands of citizen comments on the 2014 proposed rules by the Federal Communications Commission (FCC) related to net neutrality, particularly in comparison to more naive approaches to finding reused text and models that do not take into account word ordering. We also show how it finds repeated passages of text in bills introduced in Congress over a two-year period.

5.2 Related Work

5.2.1 Text Reuse Approaches

Our work is driven by interest, particularly among computational social scientists, in text reuse methods to understand large text corpora. Recent research involving text reuse includes mapping the diffusion of ideas in 19th-century newspapers during the U.S. Civil War [49], quantifying party contributions to the 2010 U.S. healthcare reform bill (Obamacare) [56], and tracing policy idea trajectories in Financial Crisis-related legislation [28]. In these domains, text reuse is a promising approach because there are substantial instances of copied or repeated text. As well, the content of the instances themselves are interpretable and often meaningful to social scientists. Determining whether text reuse exists, where it occurs in a collection of documents, and the content of repeated text are precisely the goals of both this literature and this current paper.

Currently, text reuse researchers typically use a chain of deterministic methods to find repeated sections of text in documents. In [49], for example, the researchers build a hash table of n-grams to find similar sections, run local sequence alignment algorithms to find matching pairs, and then use agglomerative clustering to group together similar passages. Each of these steps contains parameters and assumptions, such as n-gram window sizes, local sequence alignment costs, and clustering criteria, that could affect the quality of the discovered instances of text reuse. The gap that our work seeks to address is the lack of a global objective function for evaluating the quality of a text reuse solution.

5.2.2 Probabilistic Models of Text Corpora

Probabilistic generative models of text, in which words are drawn from some meaningful probability distribution over words, are a principled approach to modeling text corpora. They have at least three beneficial properties: 1) they assume a plausible, interpretable method by which the text data is generated; 2) they draw from a rich set of probability concepts and algorithms for inferring parameters; and 3) they have been shown to be empirically useful for language analysis tasks.

For instance, building a probabilistic model of text that is likely to appear in a collection of documents involves the following: 1) assuming that documents are drawn from a probability distribution over words; 2) inferring the probabilities of words by counting their relative frequencies, which can be seen as a form of maximum likelihood estimation; and 3) applying the model for prediction tasks such as speech recognition or machine translation, or visualizing the results s a word cloud, with more frequent words being more prominent.

A relevant latent-variable approach to characterizing text is probabilistic topic models, which assume that there exists a latent set of topics (multinomial distributions over words), and that each document is a multinomial distribution over topics. Popular methods for topic modeling include Latent Dirichlet Allocation (LDA) [7] and Probabilistic Latent Semantic Analysis (PLSA) [22]. Variants of these methods have been used to understand large text corpora ranging from research fields

from scientific articles [6] to political agendas in U.S. Senate press releases [17]. The highest-weighted terms in a topic often provide some sense of the "meaning" of the topic, and the corresponding topic weights for a document can indicate what the document is about.

While they have value, these word-based approaches suffers from limitations for making sense of the ideas in large document collections. First, they treat documents as unordered bags of words, losing all of the language structure and voice that inscribe meaning to pieces of writing. Second, reading the highest-probability words in a topic is not always effective at interpreting a topic model's output, a challenge that has sparked additional work in labeling topics in topic models [33, 43, e.g.] and efforts to produce more human-interpretable topics [35]. In practice, for a human to make sense of these documents and topics, it might be necessary to post-topic modeling analysis and use the highly weighted words in topics as search terms to find and read relevant documents. While there are variants of LDA that move beyond unigrams into larger n-grams, they generally do not recover clauses or longer phrases [e.g. 54, 55, 31]. For these reasons, the outputs of probabilistic topic models alone may be insufficient for making sense of the content of large document collections.

5.2.3 Text Summarization

Our task is distinct from, though arguably related to, the task of automatic text summarization, particularly in the multi-document setting [see, e.g., 38, 13, 16]. Finding frequently repeated sections of text in a large collection of documents might be useful for understanding the contents of the corpus, but we do not focus on building a written, coherent summary in this work. Identifying patterns of repeated text may itself be useful for understanding the corpus, or the repeated passages may be useful for other downstream tasks.

5.3 Probabilistic Text Reuse Model

We take a generative modeling approach and assume that our documents are comprised of text passages, which are sequences of words of varying-length "ideas" that repeat throughout the corpus. Our working hypothesis in this Probabilistic Text Reuse (PTR) model is that these ideas are useful units, both for quantitatively explaining the documents and for humans to make sense of them.

Let D be the set of N text documents, $d_1, d_2, \ldots, d_N$. Each document, d_n, consists of T_n words, $w_1, w_2, \ldots, w_{T_n}$; let V be the set of unique words, meaning that $|V|$ is the size of the vocabulary. PTR places the following generative process on the document collection D:

1. We assume that there exist I "text sequence generators" or *ideas* $\{k_1, \ldots, k_I\}$. These ideas can be of varying length. As described further below in Section 5.3, ideas are modeled as probabilistic finite state transducers (PFSTs), which are probabilistic functions over sequences of words.

2. A document d_n is a sequence of *partitions*, which contain passages of text. Each partition is either generated from a probabilistic finite state transducer (PFST) of idea k_i or from a background language model. We use z_{nm} to denote the start index of the m^{th} partition in document and a_{nm} to denote the idea or background model associated with the m^{th} partition.

In contrast to models that assume that documents are unordered bags of words, the ideas k_i consist of ordered sequences of words, which can be of varying length (generally, as shown below, ideas can be as short as clauses to paragraphs). Thus, our PTR model can capture more interpretable, meaningful entities than a standard topic model, which usually describes documents as unordered mixtures of n-grams.

Let K be the collection of ideas $\{k_i\}$, Z be the collection of partition indices $\{z_{nm}\}$, and A be the collection of assignments $\{a_{nm}\}$. We place priors $P(K)$, $P(Z)$, $P(A)$ over each of these collections; given the data D, the joint distribution $P(D, K, Z, A)$

is given by

$$\begin{aligned} P(D, K, Z, A) &= P(K, Z, A) \cdot P(D \mid K, Z, A) \\ &= P(K) \cdot P(Z) \cdot P(A) \prod_{\text{partitions, } z_{nm}} P(d_{z_{nm}} \mid k_{a_{nm}}) \end{aligned} \tag{5.1}$$

where we use $d_{z_{nm}}$ to denote the text associated with the partition z_{nm} and the assignment a_{nm} can be either an idea k_i or a background language model, which we denote by k_0.

Our objective is to find the parameter settings for K, Z, and A that maximizes the objective function, i.e.:

$$\arg\max_{K,Z,A} P(D, K, Z, A) = \arg\max_{K,Z,A} P(K, Z, A) \cdot P(D \mid K, Z, A) \tag{5.2}$$

We describe each of these factors below.

Idea Model, $Pr(K)$ In our model, we consider *frequently reused passages of text* as the ideas: Through some societal process, these ideas occur in multiple documents in our collection. We first generate the length of an idea L_i from a uniform distribution between $N_{k,min}$ and $N_{k,max}$ words and then generate each of the words a unigram language model:

$$\begin{aligned} L_i &\sim \text{Unif}(N_{min}, N_{max}) \\ k_i(l) &\sim \pi(w) \\ P(K) &= \prod_{i=0}^{I} \frac{1}{N_{max} - N_{min}} \cdot \prod_{\text{words}, w_l \in k_i} \pi(w_l) \end{aligned}$$

where $\pi(w)$ is the probability of generating word w; we set $\pi(w)$ to be the empirical probabilities of each word in the corpus. Thus, the cost of generating an idea is

the probability of generating its length and each of its words from the background language model. More frequent words in the corpus tend to have higher probabilities, so they are more likely to be part of an idea.

We seek to discover ideas that are longer phrases, as opposed to short n-grams that may occur frequently but not be as meaningful. Consequently, we set N_{min} to be about a few words long (we use $N_{min} = 5$ in our experiments. It is also worth noting that our model might consider "boilerplate" text (e.g. addresses, form language, or preambles in legal documents) as "ideas" — the definition of "boilerplate" might differ across applications. However, in practice, these boilerplate ideas should be easily identifiable from the output of the model and could be appropriately disregarded by a human analyst.

Partitions Model, $P(Z)$ We posit that the number of partitions in a document $|z_n|$ is drawn from a uniform distribution Unif$(0, N_z)$. We choose to model $|z_n|$, rather than the specific locations z_{nm}, for robustness — each partition can independently choose its length. As a result, the probability of a set of partitions $P(Z)$ is simply

$$P(Z) = (\frac{1}{N_z})^N$$

$$\begin{aligned} \text{partitions per document} &\sim \text{Unif}(1, N_z) \\ P(Z) &= \prod_{\text{documents},n}^{N} (\frac{1}{N_z - 1}) \\ &= (\frac{1}{N_z - 1})^N \end{aligned}$$

Assignments Model, $P(A)$ Each partition is assigned either to an idea k_i or the background language model with some hidden parameters θ_i:

$$P(A) = \prod_{n,m} \prod_{i=0}^{I} \theta^{\mathbb{I}(a_{nm}=i)} \tag{5.3}$$

Text Passage Model, $P(D|Z, K, A)$ A passage, d_i, is either generated from the background language model k_0 or from an idea, k_i. In this work, we simply use a unigram background language model:

$$P(d_{z_{nm}} \mid a_{nm} = k_0) = \prod_{l=1}^{L_{nm}} \pi(d_{z_{nm}}(l)) \tag{5.4}$$

where $\pi(w)$ is the empirical frequency of word w is the corpus.

Since both the partition text d_z and the assigned idea k_a are sequences of words, we need to define a conditional probability distribution of any sequence of words given an idea. Conceptually, we seek a function that does the following:

$$P(d_z \mid k_a) = \text{stochastic edit distance between } d_z \text{ and } k_a \tag{5.5}$$

We define this sequence using finite state transducers (FSTs). FSTs are finite automata that read an "input" string and, in addition, produce an "output" string. In PTR, we use probabilistic finite state transducers (PFSTs) for our ideas, which have valid probabilities as weights in each of the transitions and place a valid distribution over possible output strings. Specifically, we use the PFST formulation described in [14]. The PFST for each idea k_i is defined as follows:

- The states of the PFST are the start state q_0, states $q_1, \ldots, q_{L_i}$ corresponding to each word $w_1, \ldots, w_{L_i}$ that make up the idea k_i, and an end state q_f.

- For each of the states from q_0 to q_{L_i}, there are $|V|$ self-transitions, corresponding to insertions of any of the $|V|$ words in th vocabulary, and $|V|+1$ transitions to

the next state. The transitions are described with two symbols: one for the input string and one for the output string that are, by convention, represented with a colon (:) separation. In the case of an idea currently in the state corresponding to the word "a", the possible $2|V| + 1$ transitions can be represented as follows:

  - $a : a$ advances the FST to the next state with a match probability p_m.
  - $a : b$ advances the FST to the next state with a substitution ($p = p_s/|V|$ for each of $|V| - 1$ words in the vocabulary). There are $|V| - 1$ possible substitutions.
  - $a : \epsilon$ advances the FST to the next state with a deletion probability p_d.
  - $\epsilon : b$ keeps the FST in the current state, and represents an insertion with probability p_i. There are $|V|$ possible insertions, corresponding to each of the words in the vocabulary.

- The state corresponding to the last word, q_{L_i}, has just two types of transitions: with probability p_i, the PFST stays in the current state, representing an insertion (with probability p_i). Otherwise, the string terminates with probability $1 - p_i$.

With these operations, it is possible to generate any string from the PFST. In fact, in general, there are multiple ways to generate an output string from a PFST. We can efficiently compute the probability of an observed string—which involves summing over the probabilities of all of the paths in the PFST—through dynamic programming, i.e. the forward algorithm. For this work, we fix the parameters $\{p_m, p_s, p_d, p_i\} = \{0.8, 0.1, 0.1/|V|, 0.1\}$, which provides high weights for similar or identical strings and equal penalties for an addition, substitution, or deletion. The corresponding language model for a three-word idea is shown in Figure 5-1.

For any string, we can infer the probability that it was generated by the idea, along with most likely sequence of matches, additions, deletions, and substitutions, using a dynamic programming algorithm.

To summarize, Figure 5-2 shows the purpose of PTR: The goal is to take a large

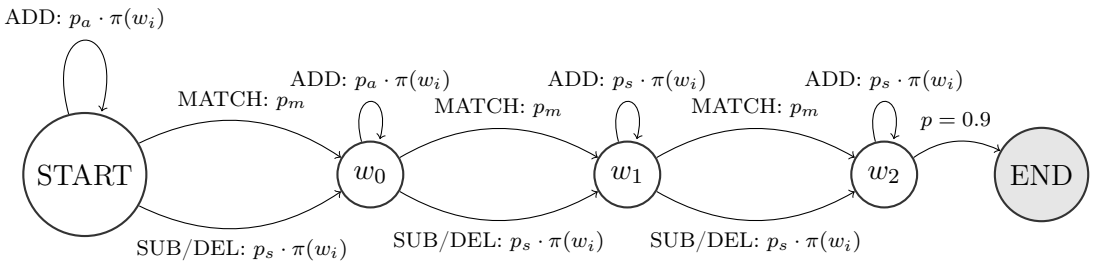


Figure 5-1: Probabilistic finite state transducer (PFST) for three-word idea

collection of unlabeled documents (D) and infer both the ideas (K) for the corpus and the partitions (Z) and assignments (A) for each document.

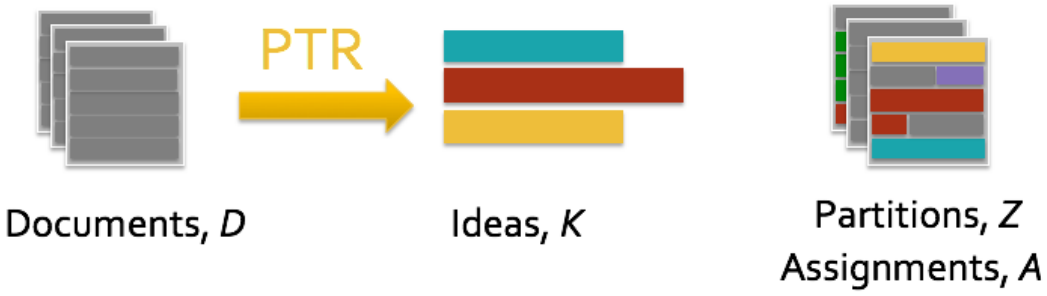


Figure 5-2: Inputs and outputs of Probabilistic Text Reuse (PTR)

5.4 Inference

Inference in our Probabilistic Text Reuse model involves inferring the ideas K, the partitions Z, and the assignment of partitions to ideas A. To obtain an approximate MAP solution to the objective in equation 5.1, we iteratively optimize the each of these three sets of hidden variables given the rest.

5.4.1 Initialization

A good initialization of the ideas k can greatly reduce the number of iterations required to converge to the approximate MAP solution. We first find a large number of word sequences that may be recurring ideas in the dataset. Through a hash table, we can count and mark the locations, which we call "anchors", of the most common

n-grams of size 5, the minimum length of an idea in our model. We choose a random subset of these anchors and perform local sequence alignment on the other anchor positions to find a common idea string. An alternative, more scalable approach is to run a sentence boundary detector on the data, and to consider sentence boundaries as passage boundaries. In a later step, we can then merge together ideas that are more than one sentence in length.

5.4.2 Updating Ideas

Given a set of partitions assigned to a particular idea k, updating the associated idea string is a form of the Steiner consensus string problem [19]. In the general case, finding the Steiner consensus string (the idea string that would minimize the distance to all of the strings in the set) is NP-hard [47]; however, choosing the best-performing representative from the set of partitions as the idea is guaranteed to be a reasonable approximation — the best performing string in the set will be no worse than twice the true optimum. We update the idea by choosing the string among those assigned to the idea that maximizes the probability of the set of partitions currently assigned to the idea.

5.4.3 Updating Partitions and Assignments

Next, we map sequences of text to our ideas or the background language model. Using dynamic programming, we find the probability that the text was generated from the PFST corresponding to each of the ideas. If this probability is greater than the the passage's score from the background language model, then this passage is assigned to the idea. Each idea now has a set of passages assigned to it that either match exactly or approximately. We devise a novel dynamic programming algorithm to accomplish this task — the alignment of an idea to a substring of a document is independent of the preceding and following substrings. As a result, it is possible to compute all possible alignments and then find the most probable path through the document.

5.4.4 Merging Ideas

Merging similar ideas means that the cost of creating one of the ideas is no longer needed, and it also increases the $P(A)$ assignment term. However, merging dissimilar ideas can reduce the likelihood, because having more ideas can match passages of text more closely. We propose the following approach to merge ideas:

1. Count pairs of consecutive idea assignments across the corpus, where "consecutive idea assignment" means that an adjoining pair of partitions are assigned to two ideas.

2. Choose the most popular consecutive assignment and propose a new idea that consists of the merged versions of the two ideas. We also retain the two original ideas, in case there are other partitions throughout the corpus that most closely align with just one of these ideas.

3. If the likelihood of the partitions belonging to these consecutive ideas increases when they are assigned to a single, merged idea, then the ideas are merged.

5.4.5 Assignment Probabilities

During inference, we fit the maximum likelihood parameters to θ_i, which results in

$$\Pr(A) = \prod_{n,m} \prod_{i=0}^{I} \frac{N_i}{|A|}^{\mathbb{I}(a_{nm}=i)} \tag{5.6}$$

where N_i is the number of passages assigned to idea i and $|A|$ is the total number of passages.

5.5 Dataset: FCC Comments on Net Neutrality

We use submissions from the first comment period (May 15 to July 15, 2014) to the U.S. Federal Communications Commission (FCC) on its proposed rules on "Protecting and Promoting the Open Internet". This collection was the FCC's largest public

comment collection to date and is publicly available. This comment period triggered enormous reaction from citizens and civil-society groups, including citizens and civil-society groups advocating for network neutrality. For example, many comments encouraged the FCC to classify and ISPs as "common carriers" (similar to telephone companies), which would, like telephone companies, empower the FCC to enforce network neutrality. Other comments, meanwhile, urged the FCC and the government not to get involved in regulations that could stifle innovation.

In general, many of comments are relatively short, often just a single sentence or paragraph. They also contain large numbers of form letters, some of which have customizable passages, prepared by civil-society groups, who mobilized individuals to submit them. We sample approximately 80,000 comments for our training set. Table 5.1 provides summary statistics of the corpus that we use to analyze our dataset.

Table 5.1: Summary of FCC comment corpus (N=800000)

mean # of words per comment (mean and standard deviation)	131 ± 2681
# of unique comments	650,300

Making sense of this large dataset of comments is a challenge for regulators and other interested parties that we seek to solve. We compare PTR with other methods in the next section.

Model Parameters We set the minimum and maximum idea lengths to be $N_{k,min} = 5$ and $N_{k,max} = 45$, and the maximum number of partitions in a document to be $N_z = 50$. These values were chosen to ensure that ideas tend to be more like sentences or sequences of sentences than single words or common multiword expressions. The PFST parameters were set to $p_m = 0.8$, $p_d = 0.1$, $p_i = 0.1/|V|$. These values were chosen to provide high weights for similar or identical strings and equal penalties for an addition, substitution, or deletion. As we describe in our discussion, future work could involve learning these parameters, perhaps on an idea-specific basis or in a tied manner.

5.6 Results

PTR finds passages of varying length and gives more accurate counts. In Table 5.2 and Table 5.3, we show an idea discovered by the sentence-boundary method with higher counts (due to finding approximate matches).

5.6.1 Noteworthy Top Ideas

Many of the top ideas come from templates from civil society groups, which encouraged people to submit form letters that they had prepared. Some examples of these ideas are below:

CREDO Action: The corpus contains 93,711 comments that include the following text: *"As an Internet user who believes strongly in the importance of a free and open Internet, I urge the FCC to reclassify broadband Internet access as a telecommunications service, and save Net Neutrality. In addition, the FCC should reject the proposed rules that would allow Internet service providers to divide the Internet into fast lanes for wealthy corporations and slow lanes for the rest of us."*[1]

Fight for the Future: 89,989 comments contain the following text: *"Net neutrality is the First Amendment of the Internet, the principle that Internet service providers (ISPs) treat all data equally. As an Internet user, net neutrality is vitally important to me. The FCC should use its Title II authority to protect it. Most Americans have only one choice for truly high speed Internet: their local cable company. This is a political failure, and it is an embarrassment. America deserves competition and choice..."*.[2]

Electronic Frontier Foundation: Approximately 68,000 comments begin with the following text: *Dear FCC, My name is Steve Roberts and I live in West Lafayette, IN. Net neutrality, the principle that Internet service providers (ISPs) treat all data that travels over their networks equally, is important to me because without it..."*.[3]

[1] See "CREDO Action: URGENT: Tell the FCC: Don't Kill The Internet", `http://act.credoaction.com/sign/fcc_nn_comments_2014`.

[2] See "Fight for the Future", `https://www.fightforthefuture.org/`.

[3] See "Electronic Frontier Foundation: Net Neutrality", `https://www.eff.org/issues/net-neutrality`.

American Commitment: Over 9,300 comments calling for less government intervention in Internet regulation (a view opposite to the ones expressed above) contained the following text: *"As an American citizen, I wanted to voice my opposition to the FCC's crippling new regulations that would put federal bureaucrats in charge of internet freedom, and urge you to stop these regulations before they're enacted...Please stop the FCC's dangerous new regulations, and protect the future of internet freedom here in America."*[4]

5.6.2 Less-Common Voices

PTR is able to discover variations of less-common voices — it aggregates passages that share a significant amount of common text. Tables 5.3 and 5.2 show examples of relatively rare ideas that, as a result of PTR's ability to capture variation, are included in the set of ideas that characterize the FCC net neutrality comments.

Table 5.2: Variations on "the internet should be open" (168 assignments)

variation on idea (sample of 21 variations)
the internet should be **publicly owned**
the internet should **never be regulated**
the internet should be **divided**
the internet should be open **and neutral**
the internet should be open **to everyone equally**
the internet should be **an** open **platform**
the internet should be **equal opportunity**
the internet should be **taxed**
the internet should be **fair**
the internet should **absolutely** be open

5.6.3 Baseline Comparisons

Topic Modeling: Table 5.4 shows the five words with the highest probability in selected topics of a 100-topic model using Latent Dirichlet Allocation [7]. The topics

[4]See "American Commitment: 808,363 Americans Tell the FCC: 'Do Not Regulate the Internet,"' `https://www.americancommitment.org/content/do-not-regulate-internet`.

Table 5.3: Variations on "keep the internet a level playing field" (244 comments)

variation on idea (sample of 90 variations)
keep the internet **an even** playing field
keep the internet **an open** playing field
keep **it** a level playing field
keep **net neutrality keep** a level playing field
keep the **net** a level playing field
keep the internet a level playing field **that it is**
keep the internet a **fair** playing field
keep the **net on** a level playing field
keep the internet **open as a fair** playing field
keep the **media landscape** a level playing field

illustrate some general themes that emerge from the corpus; however, it is difficult to directly discern submitted viewpoints from the LDA results themselves.

Table 5.4: Top ten words from selected topics of LDA model with 50 topics

topic	**keywords**
1	isps, use, slow, able, destroy
2	isps, important, new, services, better
3	communications, corporations, rules, reclassify, federal
4	free, access, equal, corporations, open
5	common, carriers, reclassify, carrier, broadband
6	companies, small, businesses, business, innovation
7	people, corporations, right, corporate, government
8	pay, content, companies, access, speed
9	wealthy, save, user, addition, believes
10	comcast, like, verizon, cable, time

Figure 5-3 shows the results of clustering documents by topic distributions. We applied k-means clustering on the document-topic distributions, then sized the nodes in the displayed plot by the number of documents in those clusters. The layout of the graph is based on a force-equilibrium approach, in which the attraction between two clusters is proportional to their similarity and a pair of clusters has a visible edge if at least one of the clusters is among the top-five closest clusters of the other. This tends to put larger and more central nodes closer to the center of the layout. As the

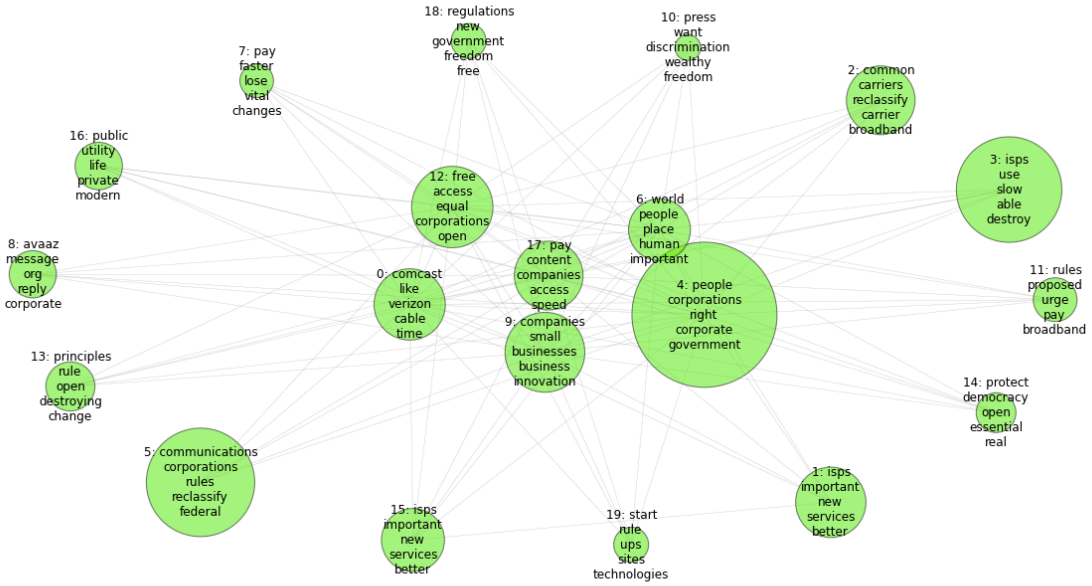


Figure 5-3: Topic-based clusters of public comments, with nodes sized by the number of comments in the cluster

figure shows, topic models can provide a useful global overview of the corpus, but they fail to capture the context in which words are used.

Common Sentences: A second approach is to run a sentence boundary detector and extract the most common sentences. As shown in Table 5.5, this process, by definition, results in coherent sentences. However, it turns out that the most frequent sentences are all from the most frequently occurring repeated comment.

5.6.4 Quantitative Comparison to LDA

One approach to quantitatively analyzing the results is to compute the likelihood of the data with respect to the model. Table 5.6 compares the log-likelihood/token for a baseline unigram language model, PTR, and a 50-topic LDA model trained on the data. Most notably, for the FCC corpus, PTR outperforms LDA on the dataset in terms of likelihood, which occurs because there is substantial text reuse — about 39% of the partitions are assigned to an idea. In contrast, while text reuse does occur in

Table 5.5: Top sentences in corpus, by frequency.

count	sentence
116,923	in addition, the fcc should reject the proposed rules that would allow internet service providers to divide the internet into fast lanes for wealthy corporations and slow lanes for the rest of us.
113,702	14-28 comments as an internet user who believes strongly in the importance of a free and open internet, i urge the fcc to reclassify broadband internet access as a telecommunications service, and save net neutrality.
111,240	title ii is the strong, legally sound way to enforce net neutrality.
111,225	this is the same trick they pulled last time.
111,225	isps are opposing title ii so that they can destroy the fcc's net neutrality rules in court.
111,221	the fcc should use its title ii authority to protect it.
111,221	please, let's not be fooled again.
111,211	it also causes tremendous economic harm.
111,211	that kills choice, diversity, and quality.
111,206	without net neutrality, a bad situation gets even worse.

the Congressional bills dataset, it represents a much smaller proportion of the corpus (approximately 1%). As a result, while the instances of text reuse are qualitatively interesting, LDA performs better at characterizing the dataset.

Table 5.6: Log Likelihood per token with unigram, LDA, and PTR models

model	log-likelihood/token
baseline unigram	-7.32
100-topic LDA	-6.48
PTR	-3.26

5.7 Discussion and Further Work

The results from PTR are better suited for making sense of this corpus than LDA or our other baseline methods. Specifically, it has the following advantages:

1. The outputs are more human-interpretable than the bags of words of other

topic modeling approaches. Our generative model is able to discover examples of text reuse throughout the corpus, cluster together similar passages of text as originating from the same idea, and find a reasonable exemplar (the "idea") that, in itself, is useful to read.

2. The model handles stop words, which, in practice, need to be removed from bag-of-words representations. In fact, stop words are an important component of PTR's representation of the data: Without them, it would be more difficult, if not impossible, to understand the passages of reused text in the comments.

3. Topic modeling does not necessarily produce topics that correspond to themes or ideas in the corpus. In LDA, for example, documents are multinomial distributions over topics, and topics are multinomial distributions over words. A topic in which few words have a relatively high amount of the probability mass could explain the data well, but it may not yield any practical significance for someone trying to make sense of a large corpus of documents. For example, topic 6 in Table 5.4 appears to be a set of words ("united", "states", "people", "fcc", etc.) that might appear very frequently throughout the corpus (but arguably should not be excluded as stop words). Similarly, while topics 1 and 3 are clearly about cable companies and Internet service providers (ISPs), the list of words poorly conveys the meaning of these statements.

4. In contrast, the ideas in the PTR model are more closely aligned to the FCC's goal of understanding the comments than the output of LDA. Finding passages of text that have high conditional probability for any of the topics is certainly possible, but it would require an additional, post-hoc analysis step. In contrast, PTR directly outputs useful-to-read ideas.

5. By assuming that passages of text are generated, with noise, from the set K, PTR identifies and finds similar statements. As a result, it provides more accurate counts or proportions of ideas than simply counting sentences, and is a more principled approach than simply clustering similar sentences afterward.

Figure 5-4 provides an illustration of how PTR can be applied to a text corpus to find key phrases: We can go directly from sequences of words (documents) directly to these PTR idea sequences. In contrast, models such as PLSA and LDA require an intermediate representation of documents, namely an unordered bag of words, which, especially in text corpora with substantial text reuse, results in a mismatch between the model and the data. In order to discover key phrases that occur in topic clusters, one needs to return to the original documents and somehow extract these key phrases. While PLSA and LDA have great utility for a general global understanding of a text corpus or for other applications, they are ill-suited for identifying commonly repeated phrases.

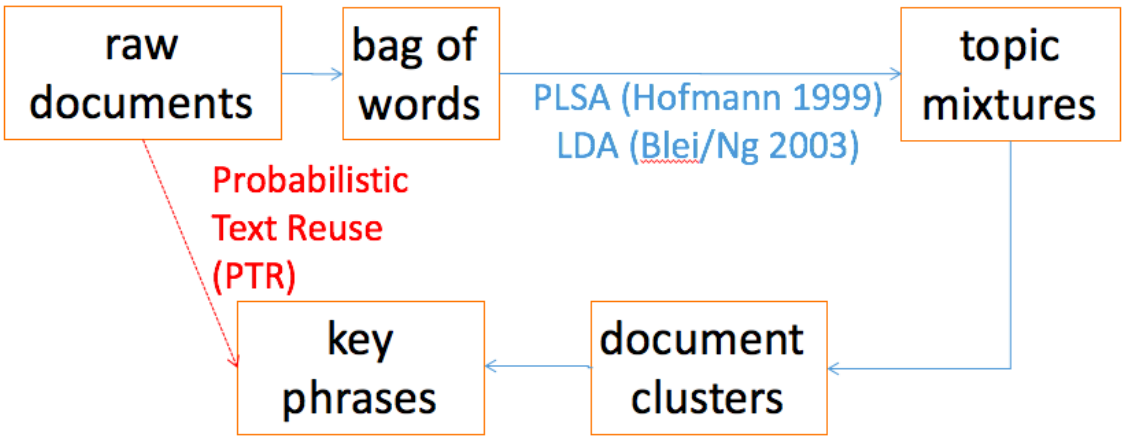


Figure 5-4: Summary of pipeline for finding key phrases with PTR vs. bag-of-words PLSA and LDA models

Future work on PTR could focus on efficiently finding the optimal set of partitions, ideas, and assignments, as well as more sophisticated background language models. More broadly, there are many interesting directions in learning the PFST parameters to go beyond text reuse and encode semantic similarity—which would be the eventual goal when extracting key ideas from large collections of text.

5.8 Conclusions

In this chapter, we introduce Probabilistic Text Reuse (PTR), a model that discovers and quantifies repeated passages of text in a large corpus of documents. On the FCC net neutrality comment dataset, we show that PTR qualitatively outperforms LDA and other baseline methods at presenting ideas from the corpus. Our approach could be useful for understanding key ideas of other large datasets, both in computational social science and in other domains.

Given the widespread prevalence of text reuse in legal and political documents, PTR could be applicable in many other large text collections. For measuring political speech, other public comment datasets collected by governments or online platforms or communities such as Facebook, Twitter, or Reddit might be useful. One could speculate that a "PTR web crawler" could find meaningful instances of significant text reuse on the Internet, perhaps bringing together online conversations or even automatically starting petitions or larger movements. More generally, these online platforms or Internet researchers might use text reuse more generally to examine how reused text propagates in their social and document networks, revealing the dynamics of how memes and other ideas spread. Meanwhile, text reuse detection could help public or private organizations break data silos (by discovering that similar text or documents are used in different parts of the organization), find new opportunities to collaborate, or even improve worker productivity, either by avoiding the duplication of the same manual work on a given document or passage of text (e.g. in language translation, customer service, or document approvals) or as an informative feature for machine learning systems. Returning to the government domain, text reuse occurs widely in bills and laws, and PTR could be useful for seeing the evolution of documents in institutions such as Congress (similar to the work in Chapter 4) or even across legislative bodies. It would be fascinating and informative, for example, to map out the diffusion of policy ideas across U.S. state lines by detecting text reuse in the bills introduced in all 50 states; going further, the influence of lobby groups (some of which are known to draft template bills) and other civil society movements might be

measured with PTR. Ultimately, written language is highly rich and flexible, allowing people to express ideas in an infinite number of ways; empirically, though, text reuse occurs in a wide range of contexts, and detecting its existence, as illustrated in this chapter, can reveal important insights into large document collections.

Chapter 6

Conclusions

This thesis presents the development and application of novel language technologies for understanding law, politics, and public policy. The projects share a common domain (legal and political documents) and technical approach (large-scale text analysis). More specifically, though, the central approach of the work in this thesis is to infer the *origins* of legal and political texts, thereby revealing information about the people or organizations behind the ideas of these documents. These hidden origins referred to authorship (Chapter 2 in the case of Supreme Court opinions, the first instance of laws (Chapter 3 in the United States Code, the introduction of policy ideas in Congress (Chapters 4, and the sources of ideas in large collections of public comments 5). By revealing these authors, sources, or origins, we discovered insights in government processes, including the dynamics of Supreme Court decision-making, the ways in which legislation is written and codified, and the nature of democratic participation through public speech in the age of the Internet.

6.1 The Role of Text Reuse in Public Data and Public Speech

Text reuse detection is a subject that reoccurs throughout this thesis: We develop techniques and applications to find repeated sections of text in the U.S. Code, bills

in Congress, and public comments on regulations. Arguably, even our approach to determining the authorship of judicial opinions can be seen as employing text reuse — we take observed n-grams to learn the justice's writing styles. The widespread existence of common legal language and the availability of different historical versions of the same document make text reuse quite common across many open government datasets; as a result, we argue that the text reuse models described in earlier chapters can be widely applicable in the legal and political domain.

Perhaps even more interesting, however, is the notion of text reuse being a form of *political speech*: The copying and pasting of text that occurs in political documents, such as legislative bills or public comments, often express explicit support for another party's point of view. In the FCC net neutrality comments, this support through text reuse occurs in at least two ways: 1) through explicit copying in the form of submitting template comments, and 2) the expression of the same idea multiple times, often with slightly different wording, which suggests that many different people have made the effort to communicate the same idea. In the same way that attendees of a political event, such as a rally or protest, may hold up copies of the same sign or shout the same slogan, members of the public can express the same idea in writing in text reuse. Discovering and measuring text reuse, therefore, can be a way to characterize the political speech in these large text corpora.

It is also noteworthy that text reuse involving wording variations of the same idea are more difficult to count and characterize. Arguably, this second pattern of text reuse may represent a more robust, popular idea — it takes more effort for individuals to express an idea in an original fashion than simply copying a template comment — but it can be more hidden in large document collections. Given that the ability to express free-form language offers one of the best ways to express diverse ideas and viewpoints, techniques that find text with similar semantic meaning seem in large collections of political speech are needed.

6.1.1 Measuring Political Speech

Our experiments with text reuse detection lead to an important question: As a democratic society, how well do we measure political speech? How effectively do our legislators and policymakers capture the viewpoints of members of the public, in order to make informed, representative public decisions? The text reuse detection methods in this thesis aim to make sense of large collections of political speech in text documents. In doing so, they center on the notion that *reused text is a form of political speech*: People "claim" speech and text from others, from short rhetorical turns of phrase to longer passages of text, to express strong agreement with that viewpoint.

More generally, it is worth considering the techniques or systems that governments use to measure political speech. In many cases, the number of people who agree with a particular viewpoint can be quantified: We can easily count votes in elections, percentages in opinion polls, or the number of signers of a petition. All of these methods, though, have limitations: They only count the number of people in agreement with specific, pre-defined ideas. Ideally, citizens should be able to contribute ideas and express themselves with the full richness of language; indeed, in some cases, like letter-writing or other communications with government officials, public forums, public comments, they have the opportunity to write or speak their views.

Figure 6-1 offers a framework for the tradeoffs between numerical and text-based approaches to measuring public speech. To begin, voting and public opinion polling are easiest to implement and evaluate, but they offer less diversity in terms of idea choices. Petitions are arguably less purely numerical and allow for more variety — they require some descriptive text that signers read and agree with. At the other end of the spectrum, public comments offer the potential for the richest diversity of opinions, but are more difficult to characterize. Techniques such as topic models, document clustering, automatic summarization, and, now, probabilistic text reuse all offer potential ways to mitigate the difficulty of analyzing and quantify the viewpoints in these large text collections.

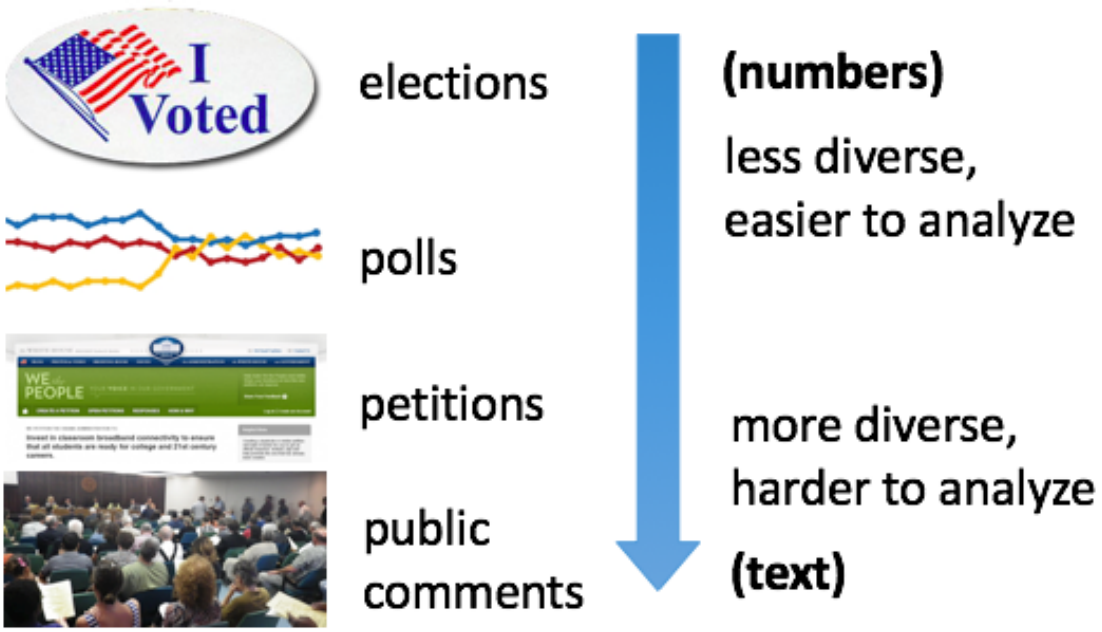


Figure 6-1: Spectrum of political speech measurement systems, from numbers to text

Thinking about these tradeoffs in political speech measurement systems leads to a related area of speculation: How might be design better systems that capture ideas. One approach might involve a hybrid of voting and free text, perhaps with the ability to propose ideas and upvote others (similar to sites like Digg and Reddit). In many ways, the large prevalence of template ideas in public comments is the result of individuals being unable to "upvote" an existing idea. Such systems also allow people to propose new ideas that could gain popularity; ideally, the number of ideas should scale appropriately to each issue. These propose-and-upvote systems also have drawbacks — websites that use them do not always result in civil, sensible, or diverse ideas, and they are just one possible design in the space of collecting text and votes. A second idea might be to incorporate discussion into such systems, allowing commenters to respond to others or engage in dialog. In this case, again, governments may be able to draw on the experiences of different online communities to understand the benefits and disadvantages of different kinds of systems.

6.2 Summary of Contributions

1. Chapter 2 describes a machine learning classifier that provides insights into the authorship of unsigned and disputed U.S. Supreme Court opinions. The classifier is trained on signed opinions of the Roberts Court to predict, with greater than 80% accuracy on a held-out test set, which author wrote an opinion. On the Obamacare decision, our model predicts that the Chief Justice wrote the majority, while the main dissent was written by Justices Scalia or Kennedy. Meanwhile, unsigned *per curiam* opinions have been predominantly written by the conservative wing of the Supreme Court in the Roberts era.

2. Chapter 3 presents a unique analysis of the United States Code through the lens of software engineering, building on the metaphor of legal code as software code. By using the text from the original codification in 1926 to the present day, our analysis examines the entire history of the U.S. Code. Our wide array of software metrics (conciseness, change, coupling and complexity) introduces a common set of techniques for describing the structure and evolution of law. Using these techniques, we discover new insights into laws, from the stark differences in network structure of reform bills and appropriations bills, as well as some of the most condition-laden parts of complex pieces of legislation like the Dodd-Frank Wall Street Reform and Protection Act of 2010.

3. In Chapter 4, we apply text reuse techniques to illustrate the trajectories of policy ideas in four key Financial Crisis bills (HERA, TARP, ARRA, and Dodd-Frank). By splitting bills up into sections and finding similar passages of text from preceding bills, we can examine, on a section-by-section basis, the "consideration time" of the policy ideas advanced in each bill. This technique brings useful patterns to the surface in the legislative process: enacted bills often contain unrelated policy riders to assemble a majority coalition of legislators to vote in favor of it, new presidential administrations often introduce entirely new text (e.g. the Obama Administration's American Recovery and Reinvestment Act contains very few provisions from the previous Congress), and different provi-

sions can have dramatically different consideration times.

4. Lastly, Chapter 5 introduces Probabilistic Text Reuse (PTR), a model that provides a global objective function for finding passages of text reuse in a corpus of documents. Rather than assuming documents are unordered sets of words, as is the case in many document clustering and classification approaches, we assume that documents consist of latent partitions, ideas, and assignments that can be learned from the dataset. These ideas are based on probabilistic finite state transducers, which assign high probabilities to the exact underlying sequence of words and assign costs to additions, deletions, and substitutions. We present methods for inferring the parameters of PTR through coordinate descent approaches. Finally, we demonstrate how PTR can be used to measure public speech in a large collection of public comments on the FCC's proposed net neutrality regulations, including large numbers of duplicate comments from civic organizations, variations of ideas, and less-common voices.

6.3 Future Work

The work in this thesis is arguably an early example of applying machine learning to the domain of legal and political texts. Consequently, there are many potential directions from both a computational and a social science standpoint. However, some compelling research questions that arise directly from this work include the following:

Authorship Attribution: Our work on Supreme Court authorship follows from a long line of work on authorship attribution, beginning with Mosteller and Wallace's seminal work on the Federalist Papers in 1963 [37]. An interesting direction would be to explicitly model the effect of law clerks in the writing process, which could help answer other interesting questions about judicial authorship; for example, it might reveal which justices rely more heavily on clerks, or whether this reliance changes over the course of their careers. A second extension could involve modeling multiple authors, as many opinions are co-signed by more than one justice (our work assumed that the main author was the label for the work). For both clerks and multiple

authors, a common modeling framework might involve assuming that documents are generated by a *mixture* of author language models, and the machine learning task would be to infer the right mixture weights for each author and document.

Law as Code: A software engineering approach to analyzing the structure and evolution of legal code is the basis for an ambitious research agenda. First, the U.S. Code is not the only "source document" that determines the law — bills, regulations, and judicial opinions also contribute to written law. Among these text corpora, at the federal level, applying software engineering metrics to the Code of Federal Regulations seems most promising — the rules are promulgated by government agencies and may have a more orderly, predictable structure that could be quantified and analyzed.

A second broad research direction for treating law as code is semantic understanding of the law: By developing semantic representations of the law, it could become possible to determine the legality of actions, identify contradictions, or identify opportunities to reduce needless complexity. In other words, semantic understanding could make it possible to "compile" the legal code, find bugs or issues, and make optimizations.

Text Reuse: Repeated text occurs frequently in legal and political documents, making text reuse a useful technique for analyzing open government datasets. Beyond the Congressional bills and public comments that were examined in this thesis, other interesting research directions include tracing repeated text across different states, which could illustrate common pathways by which policy ideas transmit geographically, or quantifying the occurrence of particular provisions in large collections of contracts. Meanwhile, the probabilistic text reuse (PTR) model introduced in this thesis could be extended to model documents more accurately. For example, a more hierarchical model might assume that "ideas" (prototypical passages of text) are generated from "topics", and that each document is a mixture of topics. Another direction could be to introduce different probabilistic models of ideas; for instance, instead of being generated from a probabilistic finite state transducer, other linguistic structures, such as parse trees, might be represented probabilistically (see, e.g. [24]). Such approaches could permit the discovery of passages with similar meanings, even if they

are not similar in a linear sense.

Appendix A

Law Is Code: Mathematical Definitions

The purpose of this appendix is to give formal definitions of the terms used in our network analysis of the U.S. Code. Because of this, it is heavy in mathematics and is intended for the interested reader. Each definition includes a description of how it is relevant to the U.S. Code and this Article.

Networks. A network $N = (V, E)$ is given by a set of vertices V (also called nodes) and a set of edges $E \subset V \times V$. We say that there is an edge from u to v (and write $u \to v$) if the pair (u, v) belongs to the set E. In our application to the vertices are sections of the U.S. Code and edges correspond to citations between sections. There is an edge $u \to v$ if and only if u cites v.

Reachability. Given two vertices u, v in a network, we say that v is reachable from u if there exists a set of vertices $x_1, x_2, ..., x_n \in V$ such that $u \to x_1 \to x_2 \to ... \to x_n \to v$. In our application, this would imply that there is a chain of citations going from u to v. Thus, any change to section v of the U.S. Code could indirectly affect section u. Throughout this section, if v is reachable by u, we say that u indirectly cites v.

Strong Connectedness. Given two vertices u, v in a network, we say that u is strongly connected to v if:

1. v is reachable from u and

2. v is reachable from u.

In this work, two sections of the U.S. Code are strongly connected if there is a path of citations via which u affects v and there is another path of citations from v to u. The simplest way in which u, v can be strongly connected is if they both cite each other. It follows that not only will changes to v affect u (because u cites v, but they can also affect itself by following a loop of citations.

Strong Connectedness as an Equivalence Relation. Strong connectedness induces an equivalence relation on the set of vertices. That is, it satisfies:

- **Reflexivity:** For any vertex v, v is strongly connected to itself
- **Symmetry:** For any u, v, we have that u is strongly connected to v if and only if v is strongly connected to u.
- **Transitivity:** For any three vertices u, v, w, we have that if u is strongly connected to v, and v is strongly connected to w, then u is strongly connected to w.

For any vertex $v \in V$, define the strongly connected component containing v as $C(v) = \{u : u \text{ is strongly connected to } v\}$. Note that, because strong connectedness is an equivalence relation, for every u strongly connected to v, we have $C(u) = C(v)$. Thus, we can partition the set of vertices V into n disjoint equivalence classes $V = V_1 \cup V_2 \cup ... \cup V_n$, where all the elements in a given equivalence class V_i are strongly connected to each other, but for $i \neq j$, any the elements in V_i, V_j are not strongly connected. Each V_i is called a strongly connected component of the network. In our legal application, the set V_i are sets of sections of the U.S. Code which all indirectly cite each other. Thus, these sets can be interpreted as a *modular decomposition* of the U.S. Code, with each V_i representing a module.

Core of a network. Given a network $N = (V, E)$, and a corresponding decomposition into strongly connected components $V = V_1 \cup ... \cup V_n$, the *core* is the largest strongly connected component, i.e. $\text{Core}(N) = \arg\max_{V_i} |V_i|$.

In our legal application, the core of the U.S. Code is a subset of sections of the U.S. Code that satisfies the following two properties:

- All sections in the U.S. Code indirectly cite each other
- The core is the largest set satisfying property (1)

Thus, the core can be seen as the largest "module" of the U.S. Code. Changing any section in the U.S. Code will, by definition, affect a large number of other sections that indirectly cite it, and is a possible way of introducing contradictions in the law, since each section in the core belongs to a large "citation loop."

Measuring the coupling of a law. When a law passed by Congress gets codified, different sections of the law become incorporated into different sections of the U.S. Code. Thus, we can interpret a given law as a subset $S \subset V$ of the sections of the U.S. Code that it is modifying. At first approximation, a law that has multiple sections in the core of the U.S. Code will indirectly affect the operation of many other laws, while a law that does not modify the core of the U.S. Code will not have such a high impact. Thus, we can approximately model how "central" a given law is by:

- Finding the set S of sections of the U.S. Code modified by the law
- Computing the size of the intersection of S with the core of the U.S. Code.

That is, given a law that modifies a set S of sections in the code, we have that its coupling metric is given by $\text{coupling}(S) = |S \cap \text{Core}(N)|$. Note that we identify the law with the set of sections of the U.S. Code it modifies. We can do this by using the Table III provided by the Office of the Law Revision Counsel.[1]

Subgraphs and the core of a given law. Given a network $N = (V, E)$ and a subset $S \subset V$ of vertices, we can define the subgraph induced by S as $N' = (S, E'(S))$ where $E'(S) = \{(u, v) \in E : (u, v) \in S\}$. That is, N' only contains elements of S as vertices, and the edges are the edges of the original network that connect nodes in S.

[1] *Table III Tool*, *supra* note 119.

This definition is useful for our work because the entirety of the U.S. code is a very large network with tens of thousands of nodes. In our work, we also find it useful to focus on individual laws. As mentioned in the above paragraph, we identify a law with the subset $S \subset V$ of sections of the U.S. Code that it modifies. This induces a subgraph $N'(S)$ which contains only the sections of the U.S. Code modified by the given law. This subgraph is frequently much smaller, with only hundreds of nodes. We define the **core of a given law** as the core of the induced subgraph $N'(S)$. A law with a large core can be interpreted as more complex and non-linear than a law with a small core, since changing one section of the law is likely to have indirect effects on a large number of other sections.

Analogously, we can define the core of a **title** of the U.S. Code as the core of the subgraph induced by all sections in that title.

PageRank. While analyzing the core of a law or title of the U.S. Code can help us understand the degree of coupling in said law or title, this type of analysis cannot be used to rank the complexity of individual sections. Because the core of a network is an equivalence class, all sections in the core are equally complex. In order to provide a ranking by complexity of sections in a piece of legislation, we use an algorithm called PageRank, which is one of the main backbones behind search engine algorithms.[2]

We first give an informal definition of the PageRank procedure to give a general intuition. Afterwards, we give a formal definition for readers with a background in linear algebra. Informally, the PageRank procedure seeks to answer the following question: if a reader followed citation links in the U.S. Code randomly, following a random citation every time they reached a new section, what is the probability that they would end up in any given section? Intuitively, sections that have a high probability of being visited by such a random walk are sections that are highly central, and which have a high indirect impact on many sections of the U.S. Code.

The PageRank algorithm uses the network's *modified transition matrix* , defined below, in order to quickly compute the probability that a given vertex will be visited

[2]Brian White, *Math 51 Lecture Notes: How Google Ranks Web Pages*, Department of Mathematics Stanford U. (Nov. 2004), `http://math.stanford.edu/~brumfiel/math_51-06/PageRank.pdf`.

by a random walk.

We now more formally give this algorithm, for readers with a background in linear algebra and algorithms. Let $N = (V, E)$ be a network with n vertices. Label the vertices of this network with the numbers 1 through n. Given two indices i, j representing vertices in V, define the following:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \text{ or if } j \text{ has no outgoing edges} \\ 0 & \text{otherwise} \end{cases} \tag{A.1}$$

$$n_j = \sum_i a_{ij} \tag{A.2}$$

$$p_{ij} = \frac{a_{ij}}{n_j} \tag{A.3}$$

Define the following matrices:

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1n} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & p_{n3} & \cdots & p_{nn} \end{bmatrix} \tag{A.4}$$

$$T = \begin{bmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \cdots & \frac{1}{n} \end{bmatrix} \tag{A.5}$$

Let $Q = \alpha P + (1 - \alpha)T$, where α is a parameter of our choice when running the algorithm. In our work, we use $\alpha = 0.85$. One can prove that there exists a unique vector $x \in R^n$ such that:

1. $x = Qx$
2. $x_i \geq 0$
3. $\sum_i x_i = 1$

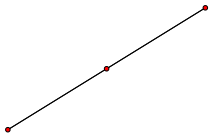
Intuitively, x is the steady state of a random walk on our given network, where the random walk resets itself with probability $1 - alpha$. This steady state is given by the eigenvector of Q with eigenvalue 1.

Appendix B

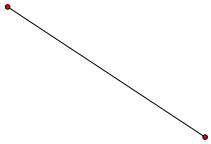
Law Is Code: Cores of Appropriations Bills

In this appendix, we show the cores of all appropriations bills passed since 1994. This visualization confirms our intuition that appropriation bills are very simple.

The definition of the core of a given law is given in Appendix A.



(a) Agriculture, Rural Development, Food and Drug Administration, and Related Agencies Appropriations Act, 1997

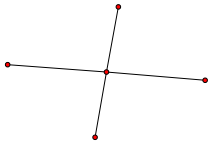


(b) Continuing Appropriations Act, 2011

(c) Continuing Appropriations Resolution, 2013

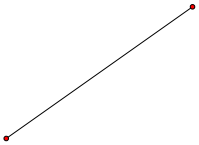
(d) Department of Defense Appropriations Act, 1999

(e) Department of Defense Appropriations Act, 2001



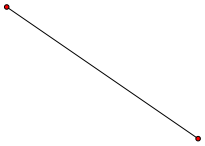
(f) Department of Defense Appropriations Act, 2003

(g) Department of Defense Appropriations Act, 2005



(h) Department of Defense Appropriations Act, 2007

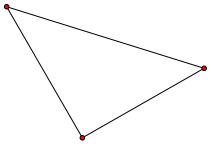
(i) Department of Defense Appropriations Act, 2008



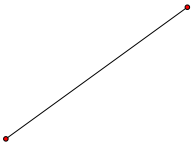
(j) Department of Defense Appropriations Act, 2010

Figure B-1: Network Representation of Cores of Appropriations Bills

(a) Department of Homeland Security Appropriations Act, 2005



(b) Department of Transportation and Related Agencies Appropriations Act, 1997



(c) Department of Transportation and Related Agencies Appropriations Act, 2001

(d) Departments of Labor, Health and Human Services, and Education, and Related Agencies Appropriations Act, 2006

(e) District of Columbia Appropriations Act, 2001

(f) District of Columbia Appropriations Act, 2005

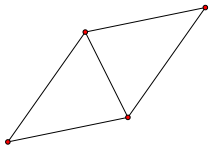
(g) Emergency Supplemental Appropriations Act for Defense and for the Reconstruction of Iraq and Afghanistan, 2004

(h) Emergency Supplemental Appropriations Act for Defense, the Global War on Terror, and Hurricane Recovery, 2006

(i) Energy and Water Development Appropriations Act, 1997

(j) Energy and Water Development Appropriations Act, 1999

Figure B-2: Network Representation of Cores of Appropriations Bills

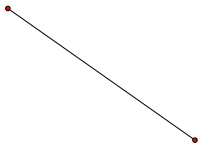


(a) Energy and Water Development Appropriations Act, 2001

(b) Energy and Water Development Appropriations Act, 2004

(c) Energy and Water Development Appropriations Act, 2006

(d) Foreign Operations, Export Financing, and Related Programs Appropriations Act, 1998



(e) Foreign Operations, Export Financing, and Related Programs Appropriations Act, 2006

(f) Legislative Branch Appropriations Act, 1997

(g) Legislative Branch Appropriations Act, 1999

(h) Military Construction and Veterans Affairs and Related Agencies Appropriations Act, 2009

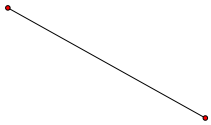
(i) Military Construction Appropriations Act, 1997

(j) Military Construction Appropriations Act, 1999

Figure B-3: Network Representation of Cores of Appropriations Bills

(a) Military Construction Appropriations Act, 2001

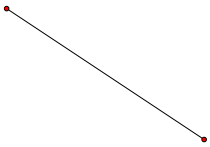
(b) Military Construction Appropriations Act, 2004



(c) Military Construction Appropriations and Emergency Hurricane Supplemental Appropriations Act, 2005

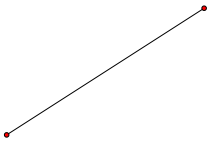
(d) Military Construction, Military Quality of Life and Veterans Affairs Appropriations Act, 2006

(e) Science, State, Justice, Commerce, and Related Agencies Appropriations Act, 2006

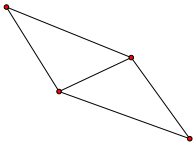


(f) Supplemental Appropriations Act, 2008

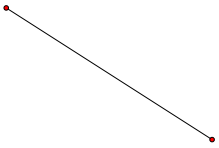
(g) Supplemental Appropriations Act, 2010



(h) Transportation, Housing and Urban Development, and Related Agencies Appropriations Act, 2010



(i) Transportation, Treasury, and Independent Agencies Appropriations Act, 2004



(j) Transportation, Treasury, Housing and Urban Development, the Judiciary, the District of Columbia, and Independent Agencies Appropriations Act, 2006

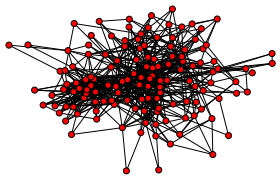
Figure B-4: Network Representation of Cores of Appropriations Bills

Appendix C

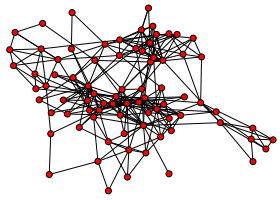
Law Is Code: Bills with large cores

In this appendix, we show the cores of all laws passed since 1994 that have a core of size larger than 50. This includes many well-known complex laws, including, for example, the Wall Street Transparency and Accountability Act of 2010. The purpose of this appendix is to illustrate instances of bills that differ from the simple core structure of appropriations bills.

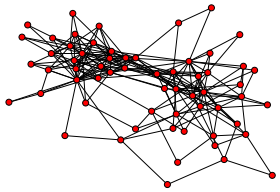
The definition of the core of a given law is given in Appendix A.



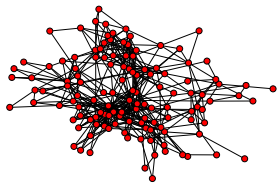
(a) Small Business Job Protection Act of 1996



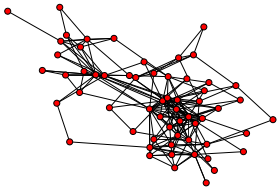
(b) Personal Responsibility and Work Opportunity Reconciliation Act of 1996



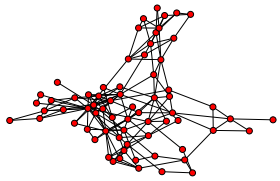
(c) Veterans Benefits Act of 1998



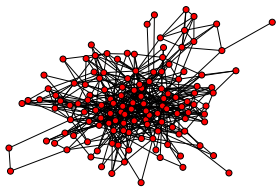
(d) TEA 21 Restoration Act



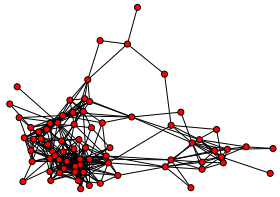
(e) Workforce Investment Act of 1998



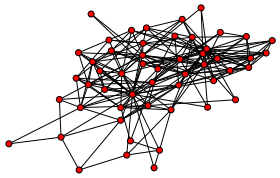
(f) Web-Based Education Commission Act



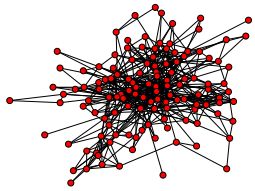
(g) Public Law 105-34



(h) Public Law 107-16

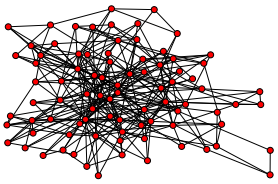


(i) Working Families Tax Relief Act of 2004

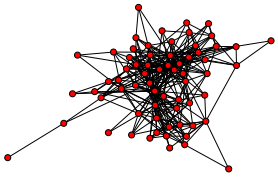


(j) Fair and Equitable Tobacco Reform Act of 2004

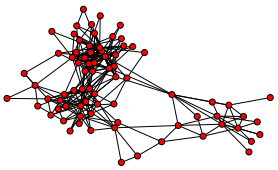
Figure C-1: Network Representation of Cores of Laws with Core of Size greater than 50



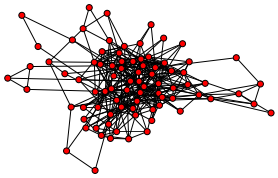
(a) Tax Technical Corrections Act of 2005



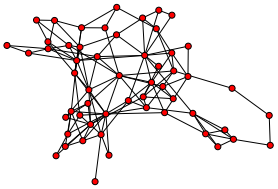
(b) Pension Protection Act of 2006



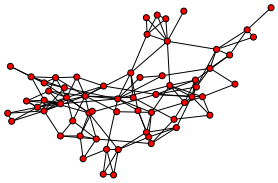
(c) Public Law 109-59



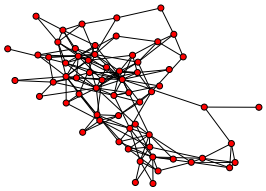
(d) Public Law 109-8



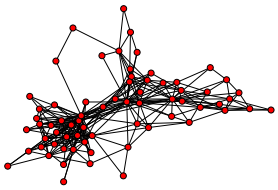
(e) Tax Technical Corrections Act of 2007



(f) Small Public Housing Authorities Paperwork Reduction Act



(g) Private Student Loan Transparency and Improvement Act of 2008



(h) Wall Street Transparency and Accountability Act of 2010

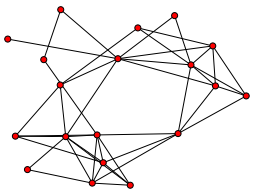
Appendix D

Law Is Code: Cores of Titles of the U.S. Code

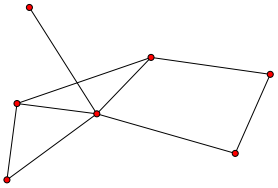
In this appendix, we show the cores of all the titles in the U.S. Code. This visualization confirms our intuition that some titles, such as Titles 13 and 14, are relatively simple while other titles, such as Titles 12, 26 and 42, are highly complex.

The definition of the core of a title is given in Appendix A.

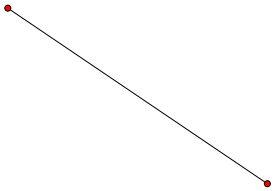
(a) Title 1



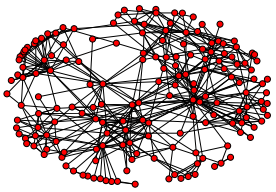
(b) Title 2



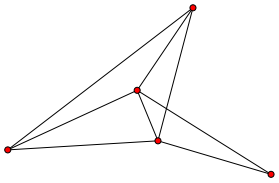
(c) Title 3



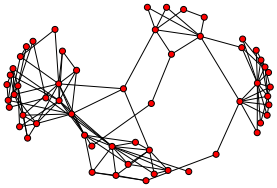
(d) Title 4



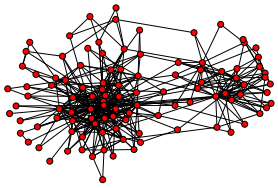
(e) Title 5



(f) Title 6

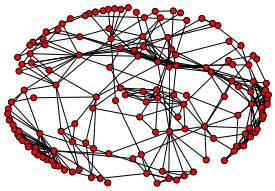


(g) Title 7



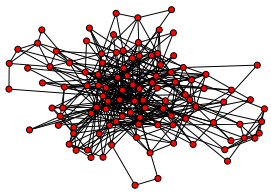
(h) Title 8

(i) Title 9

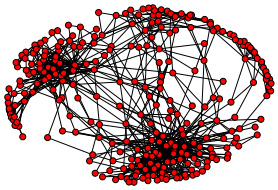


(j) Title 10

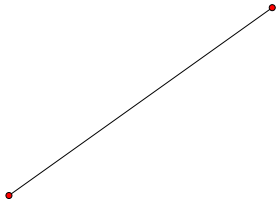
Figure D-1: Network representation of U.S. Code Titles 1 through 10



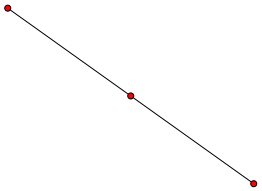
(a) Title 11



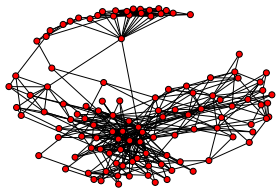
(b) Title 12



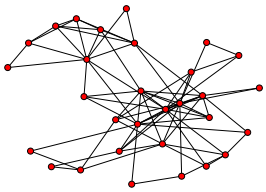
(c) Title 13



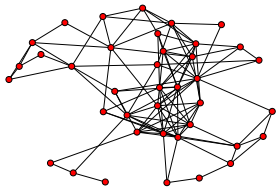
(d) Title 14



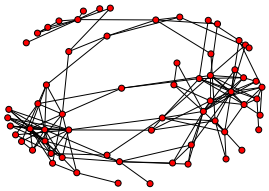
(e) Title 15



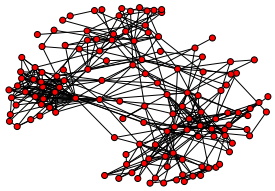
(f) Title 16



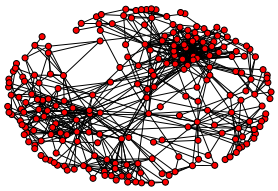
(g) Title 17



(h) Title 18

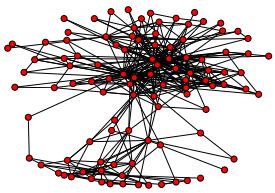


(i) Title 19

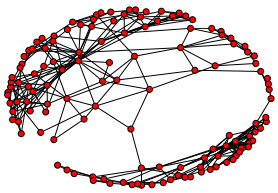


(j) Title 20

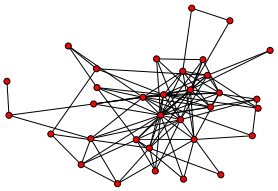
Figure D-2: Network representation of U.S. Code Titles 11 through 20



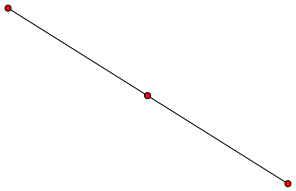
(a) Title 21



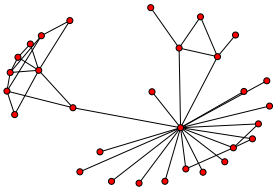
(b) Title 22



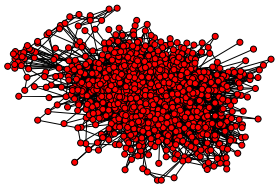
(c) Title 23



(d) Title 24

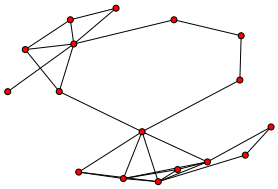


(e) Title 25

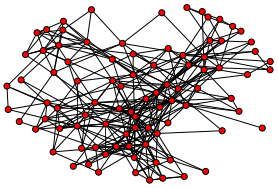


(f) Title 26

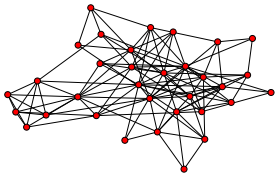
(g) Title 27



(h) Title 28

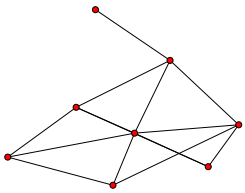


(i) Title 29



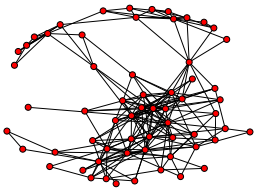
(j) Title 30

Figure D-3: Network representation of U.S. Code Titles 21 through 30



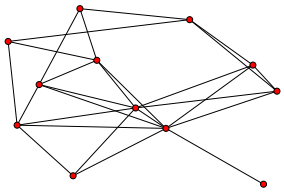
(a) Title 31

(b) Title 32

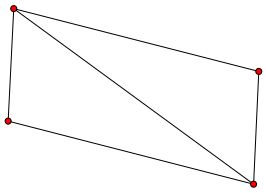


(c) Title 33

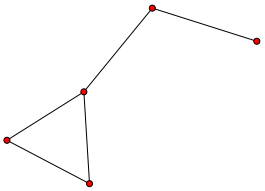
(d) Title 34



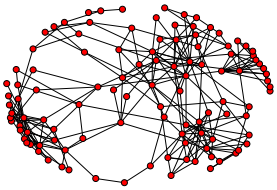
(e) Title 35



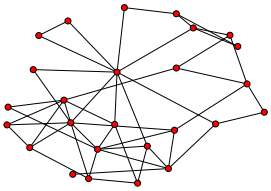
(f) Title 36



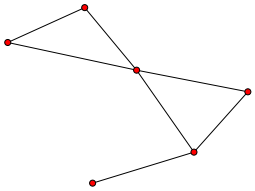
(g) Title 37



(h) Title 38

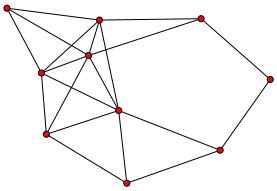


(i) Title 39

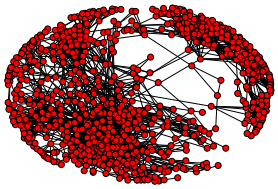


(j) Title 40

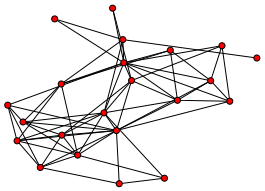
Figure D-4: Network representation of U.S. Code Titles 31 through 40



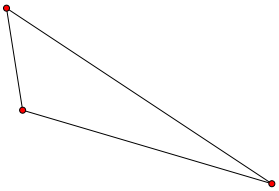
(a) Title 41



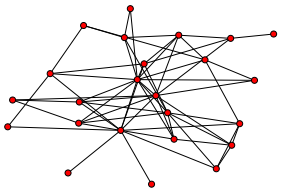
(b) Title 42



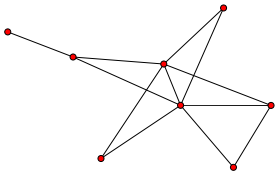
(c) Title 43



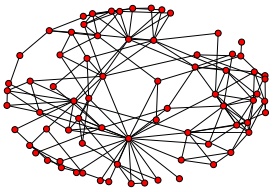
(d) Title 44



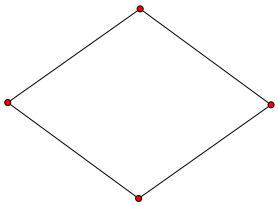
(e) Title 45



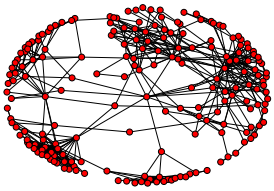
(f) Title 46



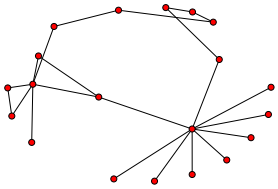
(g) Title 47



(h) Title 48



(i) Title 49



(j) Title 50

Figure D-5: Network representation of U.S. Code Titles 41 through 50

Bibliography

[1] The Annotated 8 Principles of Open Government Data. http://opengovdata.org/. Accessed: 2014-05-31.

[2] Virginia Decoded. http://vacode.org/. Accessed: 2014-05-31.

[3] Tracing Policy Ideas From Lobbyists Through State Legislatures. http://sunlightfoundation.com/tools/churnalism-us//, 2013. Accessed: 2015-12-20.

[4] E. Scott Adler and John Wilkerson. Congressional Bills Project: 1973-2014. http://congressionalbills.org/, 2014.

[5] Jaime Arguello, Jamie Callan, and Stuart Shulman. Recognizing citations in public comments. *Journal of Information Technology & Politics*, 5(1):49–71, 2008.

[6] David M Blei and John D Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, pages 17–35, 2007.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *the Journal of Machine Learning Research*, 3:993–1022, 2003.

[8] Thomas R. Bruce. Cornell Legal Information Institute. http://www.law.cornell.edu/, 2015. Accessed: 2014-05-28.

[9] Thomas R Bruce and Peter W Martin. The Legal Information Institute: What Is It and Why Is It. *Cornell Law Forum*, 20, 1994.

[10] Matthew Burgess, Eugenia Giraudy, Julian Katz-Samuels, Lauren Haynes, and Joe Walsh. Tracing Policy Ideas From Lobbyists Through State Legislatures. http://dssg.uchicago.edu/project/tracing-policy-ideas-from-lobbyists-through-state-legislatures/, 2015.

[11] Claire Cardie, Noah Smith, Anne Washington, and John Wilkerson. NLP Unshared Task in PoliInformats 2014. https://sites.google.com/site/unsharedtask2014/. Accessed: 2014-05-31.

[12] Sophie Chou, William Li, and Ramesh Sridharan. Democratizing Data Science. In *Data for Good: KDD at Bloomberg*, 2014.

[13] Janara Christensen, Stephen Soderland Mausam, Stephen Soderland, and Oren Etzioni. Towards Coherent Multi-Document Summarization. In *HLT-NAACL*, pages 1163–1173, 2013.

[14] Ryan Cotterell, Nanyun Peng, and Jason Eisner. Stochastic Contextual Edit Distance and Probabilistic FSTs. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 625–630, 2014. URL `http://aclweb.org/anthology/P/P14/P14-2102.pdf`.

[15] Sharon S Dawes. Stewardship and usefulness: Policy principles for information-based transparency. *Government Information Quarterly*, 27(4):377–383, 2010.

[16] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization-Volume 4*, pages 40–48. Association for Computational Linguistics, 2000.

[17] Justin Grimmer. A Bayesian hierarchical topic model for political texts: Measuring expressed agendas in Senate press releases. *Political Analysis*, 18(1):1–35, 2010.

[18] Justin Gross, Brice Acree, Yanchuan Sim, and Noah A Smith. Testing the Etch-a-Sketch Hypothesis: A Computational Analysis of Mitt Romney's Ideological Makeover During the 2012 Primary vs. General Elections. In *APSA 2013 Annual Meeting Paper*, 2013.

[19] Dan Gusfield. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge university press, 1997.

[20] Alexander Hertel-Fernandez. Who Passes Business's "Model Bills"? Policy Capacity and Corporate Influence in US State Politics. *Perspectives on Politics*, 12 (03):582–602, 2014.

[21] Alexander Hertel-Fernandez and Konstantin Kashin. Capturing Business Power Across the States with Text Reuse. In *Annual conference of the Midwest Political Science Association, Chicago, April*, pages 16–19, 2015.

[22] Thomas Hofmann. Probabilistic Latent Semantic Indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM. ISBN 1-58113-096-1. doi: 10.1145/312624.312649. URL `http://doi.acm.org/10.1145/312624.312649`.

[23] Mark Jickling. Fannie Mae and Freddie Mac in Conservatorship. Congressional Research Service, Library of Congress, 2008.

[24] Kevin Knight and Jonathan Graehl. An overview of probabilistic tree transducers for natural language processing. In *Computational linguistics and intelligent text processing*, pages 1–24. Springer, 2005.

[25] Michael Laver, Kenneth Benoit, and John Garry. Extracting policy positions from political texts using words as data. *American Political Science Review*, 97 (02):311–331, 2003.

[26] Michael Levandowsky and David Winter. Distance between sets. *Nature*, 234 (5323):34–35, 1971.

[27] William Li, Pablo Azar, David Larochelle, Phil Hill, James Cox, Robert C Berwick, and Andrew W Lo. Using Algorithmic Attribution Techniques to Determine Authorship in Unsigned Judicial Opinions. *Stan. Tech. L. Rev.*, 16: 503–503, 2013.

[28] William P. Li, David Larochelle, and Andrew W. Lo. Estimating Policy Trajectories during the Financial Crisis. In *NLP Unshared Task in PoliInformatics*, 2014.

[29] William P. Li, David Azar, Pablo Larochelle, and Andrew W. Lo. Law is Code: Software Engineering the United States Code. In *Journal of Business and Technology Law*, 2015.

[30] Yu-Ru Lin, Drew Margolin, and David Lazer. Uncovering social semantics from textual traces: A theory-driven approach and evidence from public statements of US Members of Congress. *Journal of the Association for Information Science and Technology*, 2015.

[31] Robert V. Lindsey, William P. Headden, III, and Michael J. Stipicevic. A Phrase-discovering Topic Model Using Hierarchical Pitman-Yor Processes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 214–222, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390948.2390975`.

[32] Alexander Madrigal. Data.gov Launches to Mixed Reviews. `http://www.wired.com/2009/05/datagov-launches-to-mixed-reviews/`. Accessed: 2014-05-31.

[33] Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. Automatic Labeling of Multinomial Topic Models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-609-7. doi: 10.1145/1281192.1281246. URL `http://doi.acm.org/10.1145/1281192.1281246`.

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[35] David Mimno, Hanna M Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 262–272. Association for Computational Linguistics, 2011.

[36] Martin Moore. Churnalism Exposed. `http://www.cjr.org/the_news_frontier/churnalism_exposed.php`, 2011.

[37] Frederick Mosteller and David L Wallace. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist Papers. *Journal of the American Statistical Association*, 58 (302):275–309, 1963.

[38] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer, 2012.

[39] Anh Phuong Nguyen and Carl E Enomoto. The Troubled Asset Relief Program (TARP) and the financial crisis of 2007-2008. *Journal of Business & Economics Research (JBER)*, 7(12), 2011.

[40] Barack Obama. Memorandum for the Heads of Executive Departments and Agencies: Transparency and Open Government. `http://www.whitehouse.gov/the_press_office/TransparencyandOpenGovernment`. Accessed: 2014-05-30.

[41] Barack Obama. Executive Order: Open Data Policy — Managing Information as an Asset. `http://www.whitehouse.gov/the-press-office/2013/05/09/executive-order-making-open-and-machine-readable-new-default-government-`, 2013. Accessed: 2014-05-30.

[42] Peter R. Orszag. Memorandum for the Heads of Executive Departments and Agencies: Open Government Directive. `http://www.whitehouse.gov/open/documents/open-government-directive`, 2009. Accessed: 2014-05-30.

[43] Daniel Ramage, Christopher D. Manning, and Susan Dumais. Partially Labeled Topic Models for Interpretable Text Mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 457–465, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7. doi: 10.1145/2020408.2020481. URL `http://doi.acm.org/10.1145/2020408.2020481`.

[44] David Robinson, Harlan Yu, William P Zeller, and Edward W Felten. Government data and the invisible hand. *Yale JL & Tech.*, 11:159, 2008.

[45] Jeffrey S Rosenthal and Albert H Yoon. Judicial ghostwriting: authorship on the Supreme Court. *Cornell L. Rev.*, 96:1307, 2010.

[46] Scout. The Sunlight Foundation. `https://scout.sunlightfoundation.com/`, 2014. Accessed: 2014-06-05.

[47] Jeong Seop Sim and Kunsoo Park. The consensus string problem for a metric is NP-complete. *Journal of Discrete Algorithms*, 1(1):111–117, 2003.

[48] Yanchuan Sim, Brice Acree, Justin H Gross, and Noah A Smith. Measuring ideological proportions in political speeches. In *Proceedings of EMNLP*, 2013.

[49] David A Smith, Ryan Cordell, Elizabeth Maddock Dillon, Nick Stramp, and John Wilkerson. Detecting and modeling local text reuse. In *Digital Libraries (JCDL), 2014 IEEE/ACM Joint Conference on*, pages 183–192. IEEE, 2014.

[50] Sunlight. The Sunlight Foundation. `https://sunlightfoundation.com/`, 2014. Accessed: 2014-05-31.

[51] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 896–903. ACM, 2005.

[52] Joshua Tauberer. GovTrack.us. `http://www.govtrack.us/`. Accessed: 2014-05-28.

[53] Kiri L Wagstaff. Machine Learning that Matters. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 529–536, 2012.

[54] Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM, 2006.

[55] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 697–702. IEEE, 2007.

[56] John Wilkerson, David Smith, and Nick Stramp. Tracing the Flow of Policy Ideas in Legislatures: A Text Reuse Approach. *New Directions in Analyzing Text as Data. London School of Economics*, 2013.

[57] Tae Yano, Noah A. Smith, and John D. Wilkerson. 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*, NAACL HLT '12, pages 793–802, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. ISBN 978-1-937284-20-6. URL `http://dl.acm.org/citation.cfm?id=2382029.2382157`.

[58] Bei Yu, Stefan Kaufmann, and Daniel Diermeier. Classifying party affiliation from political speech. *Journal of Information Technology & Politics*, 5(1):33–48, 2008.

[59] Harlan Yu and Stephen Schultze. Using Software to Liberate U.S. Case Law. *XRDS*, 18(2):12–15, December 2011. ISSN 1528-4972. doi: 10.1145/2043236.2043244. URL `http://doi.acm.org/10.1145/2043236.2043244`.