**Prepared for:**

**Centers for Disease Control and Prevention**

**Centers for Medicare & Medicaid Services Alliance to Modernize Healthcare (Health FFRDC) – A Federally Funded Research and Development Center**

Physical Activity Surveillance Modernization

Contract No. 75D30120F09743

Task Order No. 37218222

**Physical Activity Modernization Policy Technical Research Results**

**Experimental Results for Technical Research Questions**

**For Public Release**

Approved for Public Release; Distribution Unlimited 23-0609

**Version 2.0**

**April 11, 2023**

**Original Release: September 14, 2022**

The views, opinions, and/or findings contained in this report are those of The MITRE Corporation and should not be construed as official government position, policy, or decision unless so designated by other documentation. This document does not represent official views or guidance of CDC.

© 2023 The MITRE Corporation. All Rights Reserved

Record of Changes

| **Version** | **Date** | **Author / Owner** | **Description of Change** |
| --- | --- | --- | --- |
| 1.0 | 8/30/2022 | MITRE | Draft Physical Activity Modernization Policy Technical Research Results |
| 2.0 | 9/14/2022 | MITRE | Final Physical Activity Modernization Policy Technical Research Results (DNPAO feedback incorporated) |

**recom**

**Table of Contents**

[Glossary 1](#_Toc114053123)

[Executive Summary 3](#_Toc114053124)

[1 Introduction 7](#_Toc114053125)

[2 Policy Surveillance Process 7](#_Toc114053126)

[3 Data 9](#_Toc114053127)

[3.1 Background 9](#_Toc114053128)

[3.2 Data Preparation 9](#_Toc114053129)

[3.3 Experimental Data Splits 12](#_Toc114053130)

[4 Experimental Results 14](#_Toc114053131)

[4.1 Where Do Relevant Documents Exist and What Technologies Can Be Used to Pull Them into a Database in a Machine-Readable Format? 14](#_Toc114053132)

[4.2 How Do We Find Relevant Documents in a Legal Corpus with Natural Language Processing (NLP)? 19](#_Toc114053133)

[4.3 How Do We Identify Passages from Legal Documents That Answer Questions of Interest? 34](#_Toc114053134)

[4.4 How Do We Extract Answers to Questions? 39](#_Toc114053135)

[5 Recommendations 45](#_Toc114053136)

Glossary

**Application Programming Interface (API)** -– a software interface that allows one computer program to communicate with another.

**Attention** – a technique used in neural networks that attempts to imitate human attention by placing the most emphasis on portions of the input that are the most important.

**Bag-of-words –** an approach to modeling text that simply uses the terms that appear in a text and their frequencies, ignoring any other document structure or word order.

**Best Match 25 (BM-25)** – a modification of TF-IDF that accounts for document length and repeated mentions of a term.

**Bidirectional Encoder Representations from Transformers (BERT)** – a neural language model trained to predict missing words and sentences in text. In learning to perform these tasks, BERT encodes a large amount of linguistic and real-world knowledge, making it an ideal starting point for a wide range of natural language processing tasks.

**Dense Passage Retrieval (DPR)** – a document retrieval system that jointly trains vector embeddings for both passages in a document corpus and a training set of questions. At search time, the dense passage retriever encodes a vector representation for a question and identifies the passages whose vector representations are most similar to that question’s vector representation.

**Doc2Vec** – an extension of the word embedding technique applied to whole sentences or short documents. Documents that are near to one another in the vector space are expected to be similar in content.

**Corpus –** a collection of text documents. In the policy surveillance context, a corpus could be the set of all state laws and regulations.

**F1 Score –** A measure of a model’s performance on a dataset. It is defined as the harmonic mean of the model’s precision and recall and is used for evaluating information retrieval systems such as search engines and machine learning models.

**Ground truth –** A term used in statistics and machine learning that means checking the results of a machine learning model for accuracy on a given data set against what should be the true values for that data set, as labeled by human experts.

**Machine readable format –** A structured format that can automatically be read and processed by a computer. Examples include CSV, JSON, or XML, but not PDF.

**Mean Reciprocal Rank (MRR) –** A metric used in information retrieval that calculates the average rank of the desired search result in an ordered list of search results; more heavily rewarding results that are ranked higher in the search results.

**Natural Language Processing (NLP)** – a subfield of computer science focused on programming computers to process large amounts of textual data.

**Precision –** A metric used in information retrieval and classification. It is the fraction of relevant instances among all retrieved instances; the proportion of retrieved documents that are correct. In epidemiology, precision is known as positive predictive value.

**Recall –** A metric used in information retrieval and classification. It is the fraction of retrieved instances among all relevant instances; the proportion of correct documents that are retrieved. In epidemiology, recall is known as sensitivity.

**Robustly Optimized BERT Approach (RoBERTa)** – a neural language model trained to predict missing words in text. It is a modification of BERT, including more pretraining data and pretrained for more passes through the data.

**Term frequency-inverse document frequency (TF-IDF)** – measures how frequently a term appears in a document of interest when compared to how frequently that term appears in the whole document corpus.

**Textual entailment** – a natural language processing problem that tries to determine if one statement (known as the hypothesis) logically follows from another statement (the text).

**Topic modeling –** Topic modeling is an unsupervised machine learning technique that’s capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents.

**Web crawling** – the process of automatically traversing the world wide web and downloading the web pages encountered.

**Web scraping –** the process of parsing and extracting data from the web pages downloaded via web crawling.

**Word embedding –** A word embedding is a representation of a word in the form of a real-valued vector that encodes the meaning of the word. Words that are closer in the vector space are expected to be similar in meaning.

Executive Summary

This report describes the results of a series of experiments conducted by the Centers for Medicare and Medicaid Services Alliance to Modernize Healthcare (Health FFRDC) operator, The MITRE Corporation (MITRE) for the Division of Nutrition, Physical Activity, and Obesity (DNPAO), at the CDC National Center for Chronic Disease Prevention and Health Promotion (NCCDPHP).

The experiments examine how well machine learning and artificial intelligence technologies can augment the various steps of the policy surveillance process. Policy surveillance involves the ongoing systematic *collection, analysis, and dissemination* of information about laws and other policies across jurisdictions over time.

The datasets that were used in the experiments, the experimental research questions MITRE set out to answer and a brief summary of results from multiple experiments are described below.

**Datasets**

To evaluate the results of our experiments, we compared our results to previously completed policy surveillance studies related to physical activity. We used the Complete Streets dataset and the Local Inclusionary Zoning dataset described below, both developed by Temple University’s Center for Public Health Law Research (CPHLR). We selected these datasets in collaboration with both DNPAO and CPHLR, with the goal of replicating one state level and one local level dataset, as well as examining datasets related to both transportation and land-use policies.

1. Complete Streets (<https://lawatlas.org/datasets/complete-streets>). The Complete Streets policy surveillance study analyzed state-level transportation laws and policies. The dataset includes statutes, regulations, and policies in all 50 U.S. states and the District of Columbia through July 1, 2020; 25 of 51 have Complete Streets policies. This study includes the answers to 13 questions for each state and provides the specific text in the law used to arrive at the answer to each question.

2. Local Inclusionary Zoning (<https://lawatlas.org/datasets/inclusionary-zoning>). The Local Inclusionary Zoning policy surveillance study analyzes local zoning policies for 10 selected cities. This study includes the answers to 14 questions for each city and provides the specific text in the law used to arrive at the answer to each question. The dataset includes laws in effect as of December 1, 2018. The jurisdictions include: Boulder, CO; Burlington, VT; Cambridge, MA; Evanston, IL; Irvine, CA; Highland Park, IL; Lake Forest, IL; San Diego, CA; Santa Fe, NM; and Stamford, CT.

**Experimental Results**

1. *Where do relevant documents exist and what technologies can be used to pull them into a database in a machine-readable format? (Data Collection)*

One needs to search through state and local laws and regulations and state-level Department of Transportation policy documents to find the relevant laws and policies addressing Complete Streets and Local Inclusionary Zoning. Justia (law.justia.com) hosts the laws and regulations for all 50 states and DC in a regular format that is highly amenable to automated web crawling and extraction; however, this process is time consuming and needs to be planned for months in advance. Local/municipal laws are stored across multiple databases; researchers need to specifically identify the site hosting the laws for the jurisdictions of interest.

1. *How do we find relevant documents in a legal corpus with natural language processing? (Document Retrieval)*

We experimented with two approaches to identify relevant documents. The first approach relies on the human researcher identifying an example law, and then continuing to find documents similar to that example. Techniques that follow this approach include Doc2Vec and using topic modeling to find documents with similar topic signatures. These approaches were effective at identifying documents with similar content. The second approach does not require a researcher to have first identified a sample document, but instead seeks to identify relevant documents based solely on the question the researcher is trying to answer. Techniques we experimented with here include TF-IDF, BM-25, and Dense Passage Retrieval. We found they were not particularly successful in this application, likely because the policy surveillance datasets are too small to generate sufficient training data.

1. *How do we identify relevant passages from legal documents that answer questions of interest? (Clause Identification)*

We experimented with both pretrained and fine-tuned neural question-answering systems to see how well they do at identifying the specific sentence(s) in the text that answer the question. We found that the question-answering systems can achieve moderate accuracy if they are trained for many passes through the data. However, we do not think their performance rises to the level that would be useful to a policy surveillance researcher, and we believe similar results can be achieved by using Doc2Vec search techniques on short excerpts from longer legal documents.

1. *How do we extract answers to questions? (Answer Resolution)*

Given the sentences extracted from laws, policy surveillance seeks to arrive at a concrete answer to a research question, which is typically either binary or a categorical answer that can be converted to a binary format. We experimented with two techniques: textual entailment models, which determine if a hypothesis logically follows from a text, and training various binary classifiers to determine if the text represents a “yes” or a “no” answer. The textual entailment models did not work well on this legal data, while the binary classifiers based on BERT and RoBERTa performed very well.

**Recommendations**

Figure 1 illustrates the steps of the policy surveillance process, indicating which steps would be augmented by particular technologies. Recommendations resulting from the experiments are outlined below the figure.

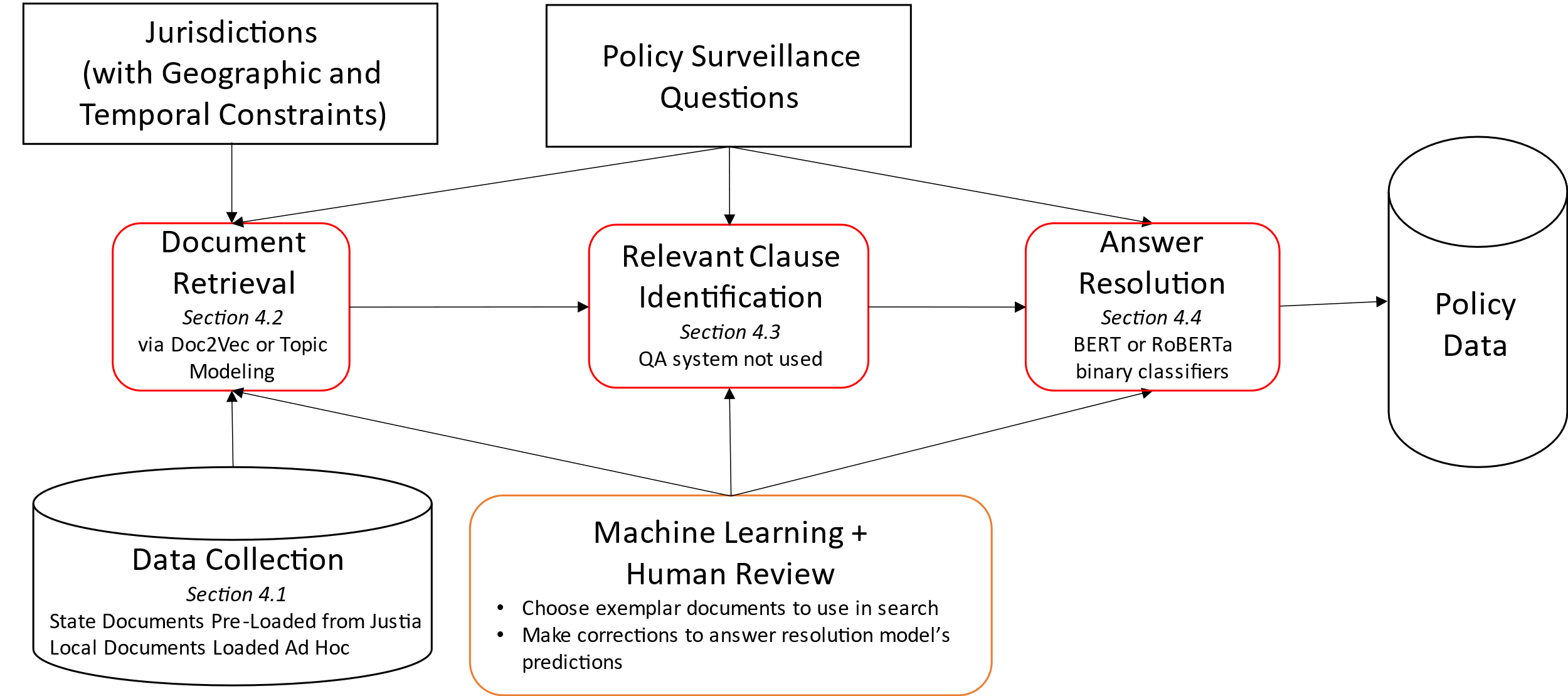


Figure 1: Policy Surveillance Process with Recommendations of How to Incorporate Artificial Intelligence

**Maintain a database of state-level statutes; Obtain local statutes on an ad hoc basis**

In preparation for a policy surveillance study, the relevant legal documents would need to be gathered. A computer program could automatically crawl Justia and parse the documents to populate a database of state codes and regulations. The code for municipal jurisdictions, as well as any state-level legal documents that are not part of the state code (such as policy documents from state agencies) would need to be individually acquired and processed. These documents would then be automatically segmented into 100 to 200-word excerpts, and a document vector model (or potentially a topic model) would be trained over these excerpts.

**Utilize document embeddings to identify relevant documents; continue experimenting with using a topic-model based search**

Once complete, a policy surveillance researcher could start searching for the relevant legal documents using standard keyword search techniques. The researcher could augment their initial search by using word embeddings trained on the legal documents to identify more search terms that might point to additional relevant documents. Once the researcher has identified the relevant documents for an initial jurisdiction, they can use the document vector model (or potentially the topic model) to identify documents with similar content. This could be used to find additional documents associated with the original jurisdiction of interest, or to find similar legal documents in different jurisdictions. While the documents identified will be 100 to 200-word excerpts of the complete legal text, one can envision a user interface that would allow the researcher to read the excerpts in their larger context.

**Convert all policy surveillance questions to binary format and use BERT and RoBERTa-based binary classifiers to perform answer classification.**

The system could then combine this legal text with the text of the specific policy surveillance question of interest (provided by the policy surveillance researcher) and apply a BERT or RoBERTa-based classifier to predict a yes or no answer to that question. The BERT or RoBERTa-based classifier would have been trained in advance on a large number of previously completed policy surveillance studies. The policy surveillance researcher would review the answer predicted by the system for correctness. If the system was incorrect, the policy surveillance researcher could indicate as such and the classifier could be retrained, improving the predictions for future policy surveillance studies.

1. Introduction

The Division of Nutrition, Physical Activity, and Obesity (DNPAO) at the CDC National Center for Chronic Disease Prevention and Health Promotion (NCCDPHP) is interested in modernizing their approach to policy surveillance as part of an overall effort to enhance and modernize physical activity surveillance data collection. Broadly, policy surveillance is a recent and complex field requiring legal expertise, and the process of conducting policy surveillance is labor-intensive. The evolving field of machine learning offers potential applications to policy surveillance that could assist in making the overall process more efficient.

As part of its efforts, DNPAO has tasked the Health Federally Funded Research and Development Center (FFRDC), operated by The MITRE Corporation (MITRE), with identifying components of the policy surveillance process that are amenable to automated methods.

This report outlines the results of a series of experiments to determine how well machine learning and artificial intelligence technologies can automate or augment the various steps of the process of policy surveillance. Section 2 provides background on how the policy surveillance process can be broken down into discrete steps for automation or machine augmentation and lays out how these steps informed the technical research questions we sought to answer. Section 3 provides information on the existing policy surveillance datasets we used to explore these research questions and describes how these datasets were processed into suitable formats for experimentation. Section 4 details the results of experiments performed to answer the technical research questions. Finally, Section 5 outlines how the lessons learned from this experimentation might be applied in future policy surveillance work.

1. Policy Surveillance Process

Policy surveillance involves the ongoing systematic *collection, analysis, and dissemination* of information about laws and other policies across jurisdictions over time.[[1]](#footnote-2) The policy surveillance process involves:

* accessing collections of diverse legal documents
* searching them for those relevant to the topic of interest
* identifying the text that answers the questions in the policy surveillance study
* compiling the binary or categorical answers from that text into a dataset.

Figure 2 shows an illustration of the steps of the policy surveillance process.

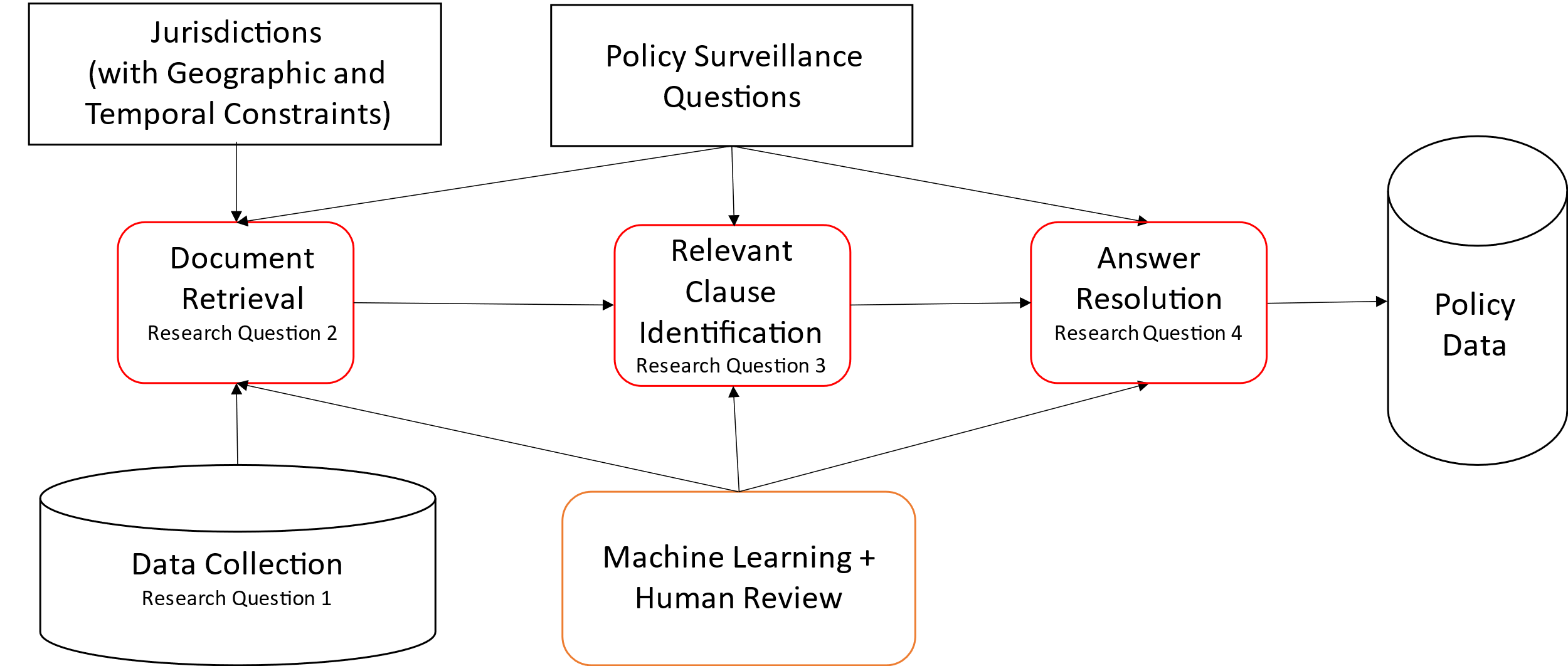


Figure 2: Policy Surveillance Process with Breakdown of Steps to Transform Policy Text to Data

To best augment future policy surveillance efforts, it is paramount to understand where in the process automated methods perform well and where they do not. Separating these tasks into individual research questions helps us gain an understanding of which parts of the process automated approaches work best and which parts require human knowledge and intervention. Thus, the four research questions we sought to answer are as follows:

1. Where do relevant documents exist and what technologies can be used to pull them into a database in a machine-readable format? (Data Collection)
2. How do we find relevant documents in a legal corpus with natural language processing? (Document Retrieval)
3. How do we identify relevant passages from legal documents that answer questions of interest? (Clause Identification)
4. How do we extract answers to questions? (Answer Resolution)

In the following sections, we identify the datasets we experimented with and discuss the results of multiple experiments to identify the best way to answer these four research questions.

3 Data

3.1 Background

We have chosen to apply natural language processing to replicate two policy surveillance datasets developed by the Center for Public Health Law Research (CPHLR) Policy Surveillance Program at Temple University using fully manual, traditional methods:

* The Complete Streets study[[2]](#footnote-3) and
* the Local Inclusionary Zoning study[[3]](#footnote-4).

These two datasets were chosen in consultation with DNPAO and CPHLR and will provide insights into different aspects of laws that promote physical activity (Table 1). For each jurisdiction, the datasets identify a categorical or numeric answer to each policy surveillance question, the legal document containing that answer, and the extracts and pincites[[4]](#footnote-5) from the documents providing the specific evidence for each question’s answer.

Table 1. Characteristics of CPHLR’s Complete Streets and Local Inclusionary Datasets

|  |  |  |
| --- | --- | --- |
|  | **Complete Streets** | **Local Inclusionary Zoning** |
| **Level of government** | State level | Local level |
| **Subject matter** | Transportation policy | Land use policy |
| **Jurisdictions included** | 50 States and DC | 10 select cities |
| **Number of questions in dataset** | 13 | 14 |

Our goal is to see how well we can use artificial intelligence and machine learning technologies to identify the same extracts of legal text and see how successful natural language processing techniques are at processing that legal text to arrive at the same answers to the policy surveillance questions as produced by the manual policy surveillance process.

3.2 Data Preparation

Using a Google Chrome internet browser, we inspected the source code of the Policy Surveillance Program’s websites for the Complete Streets and Local Inclusionary Zoning laws studies and selected a machine-readable JSON[[5]](#footnote-6) extract containing the text of the policy surveillance research questions, the answers to the research questions, and the text of the extracts and pincites used to answer each question. This JSON contained extra information related to the website layout and often used hash values[[6]](#footnote-7) to represent question text and law titles. Therefore, we processed this JSON into a format suitable for further experimentation which contained only the necessary information and used free text rather than hash values. In the JSON found on the CPHLR’s website, sometimes the text of the extract used to answer a question was listed as “Full Text” rather than the actual legal text, indicating the entire law was considered relevant to answering the question. In these instances, we replaced “Full Text” with the actual text of the law in question.

To explore the document retrieval and question answering research questions, we need to identify relevant documents and passages regardless of whether the policy surveillance question ultimately is answered with a “yes” or a “no”. However, in the Law Atlas policy surveillance studies, supporting legal text was provided only when a question had a “yes” answer. To determine the relevant legal text indicating that a question’s answer was “no”, we manually read through the legal text CPHLR used for that jurisdiction and identified the relevant passage ourselves.

**Data Recommendation**: When practical, policy surveillance practitioners conducting policy surveillance manually should indicate the legal text supporting “no” answers as well as those supporting “yes” answers.

When CPHLR indicated that a law’s full text was used to answer a question, this answer passage was quite lengthy and usually contained more information than was strictly needed to answer the question. Even when an extract was provided, the extract was sometimes also longer than needed. For example, in the Local Inclusionary Zoning study, CPHLR identifies the following passage to support that the answer to the question “What is the period of affordability for rental units” for Lake Forest, IL is “in perpetuity” (emphasis added):

“(B) Rental of Affordable Housing Units. **In Covered Development Projects that contain rental units, Affordable Housing Units shall be rented to Low- and Moderate-Income Households in perpetuity, or as long as permissible by law.** The owner of the Covered Development Project shall execute and record all documents required by Section 20A-12 of this Chapter to ensure compliance with this Section. In each case, the rental Affordable Housing Unit shall be occupied by a Low- or Moderate-Income Household, and such unit may not be leased or subleased unless expressly approved by the City and such lease or sublease is to permit occupancy by a Low- or Moderate-Income Household. (1) In the event that the owner of a Covered Development Project with one or more rental units sells the development, the new owner shall be required to continue to provide the Affordable Housing Units in accordance with this Chapter. (2) If the owner of a Covered Development Project with one or more rental units converts the development to condominiums, the development shall be subject to the for-sale development requirements of this Chapter.”

While the entire passage provides useful context, only the bolded sentence is necessary to provide a categorical answer to the policy surveillance question. To address memory constraints associated with some approaches to question answering and answer classification, and when the extract identified to support the answer to the question was longer than two sentences, we manually identified the specific one or two sentences in the longer extract that directly supported the question. Thus, in our final dataset we had both a “long answer text” and a “short answer text” specified for each question (in some instances these answer texts were identical).

**Data Recommendation**: If policy surveillance practitioners wish to incorporate AI to assist in completing policy surveillance studies, we suggest they identify both the most concise extract of legal text that is necessary to provide a simple binary or categorical answer to a question and the full legal text providing greater context. The more concise extract can be used to support training AI methods, while the full text can help the researchers identify caution notes or describe the policy in more detail than conveyed in the simple answer.

Policy surveillance questions can be classified as binary (yes/no), categorical – mutually exclusive (choose one answer from three or more answer choices), or categorical – select all that apply. With the diversity of questions that could be asked in policy surveillance, it is unrealistic to train an automated system to answer one specific question or to train separate models for each individual question; we need a system that can generalize to multiple questions. Therefore, for our answer classification experiments, we experimented with using one classifier to answer any binary question. To make a “categorical – select all that apply” question fit into this paradigm, we converted each categorical question into multiple binary questions, with one binary question corresponding to each possible answer choice.

* For example, Question 11 in the Complete Streets study, “What type of projects trigger the policy?”, which has three answer choices of “New construction projects”, “Maintenance projects”, and “Project types not specified in the law”, can be converted into three binary questions: “Do new construction projects trigger the policy?”, “Do maintenance projects trigger the policy?”, and “Do project types not specified in the law trigger the policy?”, with the answer “yes” if the option was selected and “no” if it was not.

We used the legal text identified by CPHLR to support the answer to the categorical question as the supporting legal text for all derived binary questions. Depending on the question, the short answer text for each derived binary question may vary if different sentences in the longer extract provided support for different categorical answer options.

One of our experiments in answer classification involved using a textual entailment model, where we determine if one sentence logically follows from another sentence. In this instance, we wanted to identify if a “yes” answer to a policy surveillance question logically follows from the identified relevant text. To use a textual entailment model, we need to rewrite the policy surveillance questions into statements. All questions in both policy surveillance studies were rewritten as statements, and we kept both the original question and the statement form in our final dataset.

* For example, Question 5 in the Complete Streets study, “Must information about Complete Streets implementation be publicly released?” can be written as “Information about Complete Streets implementation must be publicly released.”

These textual conversions were done using a combination of automated and manual methods. We manually read the policy surveillance questions and looked for patterns in how to convert them to binary questions and/or statements. Then we wrote code that started with the policy surveillance question as it was originally written and applied the correct conversion pattern.

In some of our experimentation, we wanted to see if results were improved by including the jurisdiction of interest in the policy surveillance question. Therefore, in our dataset, we included a version of the question as it was written in the original policy surveillance study and a version of the question including the jurisdiction. In most cases, this simply meant appending “in [jurisdiction]” to the end of each question.

* For example, in addition to the question “Is the policy mandatory for developers?”, we also included the question in the form “Is the policy mandatory for developers in Burlington?”.

Thus, for each question in the original policy surveillance study, our final dataset included the following elements:

* The question text as it was written in the original policy surveillance study
* The question converted into multiple binary questions, if applicable
* The question text with the jurisdiction included
* The question text rewritten as a statement
* The binary answer to the question
* The legal text supporting the answer to the question, as identified in the original policy surveillance study for “yes” answers, and as manually identified by us for “no” answers (the long answer text)
* A shorter excerpt we identified from the legal text as directly supporting the answer to the question (the short answer text)
* The full 200-word passage containing the legal text supporting the answer to the question, for evaluating document retrieval (see section 4.2.2)

3.3 Experimental Data Splits

For our experimentation, we divided both the Complete Streets and Local Inclusionary Zoning datasets into a training portion, a validation portion, and a testing portion. Dividing data into training, validation, and testing sets is known as splitting the data.

We used:

* The training portion as input for training models
* The validation portion to choose among the best trained models
* The testing portion to evaluate how well the models perform on completely novel data

The Complete Streets dataset includes answers to questions for 25 states. (All 50 states were examined, but only 25 had Complete Streets policies.) To test how much data was needed to train a well-performing model, we created two different data splits: one with five states in the training set and one with ten states in the training set. We used the same validation and testing sets for both data splits so we could directly compare results. Since the Local Inclusionary Zoning dataset only includes ten jurisdictions, there was not enough data to create unique validation and testing sets. For this dataset, we chose five jurisdictions for the training set and five jurisdictions for the validation/testing set. Using a random number generator, we assigned the following jurisdictions to each data split, shown in Table 2:

Table 2. Data Splits for Experimentation

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **Training** | **Validation** | **Testing** |
| **Complete Streets – small dataset** | California, Florida, Iowa, Minnesota, Nevada | Delaware, District of Columbia, Maine, Vermont, West Virginia | Colorado, Connecticut, Hawaii, Indiana, Maryland, Massachusetts, North Carolina, New Jersey, Rhode Island, Tennessee, Utah, Washington |
| **Complete Streets – large dataset** | California, Florida, Georgia, Iowa, Louisiana, Michigan, Minnesota, Nevada, New York, Virginia | Delaware, District of Columbia, Maine, Vermont, West Virginia | Colorado, Connecticut, Hawaii, Indiana, Maryland, Massachusetts, North Carolina, New Jersey, Rhode Island, Tennessee, Utah, Washington |
| **Local Inclusionary Zoning** | Boulder, Highland Park, San Diego, Santa Fe, Stamford | Burlington, Cambridge, Evanston, Irvine, Lake Forest | Burlington, Cambridge, Evanston, Irvine, Lake Forest |

4 Experimental Results

For each research question outlined in Section 2, we describe the methodology for the experimentation and the results for both the Complete Streets and Local Inclusionary Zoning datasets.

4.1 Where Do Relevant Documents Exist and What Technologies Can Be Used to Pull Them into a Database in a Machine-Readable Format?

For a policy surveillance study, researchers in essence must search through the entire set of statutes, regulations, and policies for each jurisdiction of interest to find the documents relevant to their topic.[[7]](#footnote-8) This exact search is what occurs when a user enters a query into a commercial legal search engine such as Westlaw, LexisNexis, or Municode. To replicate this search process with machine learning, we would need to be able to store the entire contents of all statutes, regulations, and policies for each jurisdiction of interest. However, it is often not feasible to download a jurisdiction’s entire body of code using a legal search engine, and what you can download from a legal search engine is not easily machine-readable. We have, however, identified some alternative resources that will allow us to obtain a jurisdiction’s entire legal code in a machine-readable format. Different resources are available for state and local jurisdictions and are discussed in the following subsections.

4.1.1 State-Level Statutes and Regulations

Commercial legal search engines such as Westlaw and LexisNexis allow users to search through the entire body of state-level statutes, regulations, and policies. However, while Westlaw and LexisNexis allow users to download specific search results, it is not feasible to download a state’s entire body of code through these services. Furthermore, when one downloads results from these search engines, the documents returned are typically either Microsoft Word documents or PDF documents. These document formats are easily readable to a human but are not machine readable. To perform natural language processing on these legal documents, we would need to extract just the raw text from the documents (eliminating most information pertaining to font and layout that is not relevant to interpreting the meaning of the text) and store any metadata of interest (such as the document title and date) in a structured way. To adequately replicate the policy surveillance process, we need an alternative source to bulk download all state-level statutes and regulations and save the text in a machine-readable format.

The company Justia provides the text of the state codes and regulations for all 50 states and the District of Columbia at <https://law.justia.com>. For the state codes, Justia provides the code for the most recent complete calendar year (currently 2021) and historical versions for each state dating back as far as 2005 (the exact historical years available vary from state to state). Justia does not provide an API to access the code, nor does it provide a method for bulk downloading statutory text. However, Justia stores every state’s statutes across multiple web pages with predictable URLs and uses the same HTML template to display the text of each law. This makes it very easy to write a program to automatically traverse the Justia website (known as web crawling), parse the HTML page containing each law, and access every statute’s title and text. State regulations are similarly available; however, Justia only hosts the most recent complete calendar year’s version of state regulations and not historical versions.

Downloading the state codes from Justia is a time-consuming process. In order to avoid overwhelming Justia’s servers, we must specify a time delay between each query for a webpage; a MITRE expert recommended specifying a time delay of 10 seconds between each query; therefore, it takes a long time to download the entire code for every state. After two weeks of crawling, the web crawler had not yet gathered the code for just the state of California. While California has one of the larger state codes in the country, it was still not feasible to use Justia to download state codes for these experiments. However, with advance planning crawling the Justia website to obtain the complete state codes for all jurisdictions of interest in future work is feasible.

Fortunately, we were able to use other resources to get state laws for some jurisdictions to use in our experimentation. Public.Resource.Org[[8]](#footnote-9) provides the regulations for all 50 states in a bulk download in XML format. For state statutes, Public.Resource.Org provides the laws for 10 states[[9]](#footnote-10) in a common HTML format.

Individual states vary in the support they provide for programmers to access their state statutes in a machine-readable format. Massachusetts, New York, Utah, and Virginia provide Application Programming Interfaces (APIs) that allow one to download state statutes in JSON format.[[10]](#footnote-11) A programmer could easily extract the legal text from the results of these API calls but would have to learn a different API for each state. Most other states provide web interfaces to allow a user to read the text of their statutes[[11]](#footnote-12), and perhaps download the text of a particular statute in Microsoft Word, PDF, or HTML, but do not provide resources for bulk download of all statutes.

For our experimentation on the Complete Streets dataset, we used Public.Resource.Org to download the state regulations for all 50 states and the statutes for the 10 states Public.Resource.Org has processed. Additionally, we used the APIs for Massachusetts and New York to download and process the text of those state statutes. We used just those two states’ APIs since both Massachusetts and New York have state laws addressing Complete Streets policies, whereas Utah’s Complete Streets policy is not found in its state code. We did not need to use the Virginia API since its statutes can be accessed through Public.Resource.Org.

For the remaining 38 states and the District of Columbia, we used the LexisNexis search engine and manually ran the same search query found in CPHLR’s Complete Streets coding protocol: “complete streets” OR “all ages” OR “all abilities” OR “accommodates all users”. We downloaded the results as Microsoft Word documents.

In the Complete Streets policy surveillance study, researchers identified 22 relevant policies from 20 states that were not found in state statutes or regulations, but instead were found in policies issued by state agencies such as Departments of Transportation. These types of policies are not stored in legal search engines such as LexisNexis, and we are not aware of any comprehensive resource containing these types of policies. CPHLR identified these documents by visiting each state’s Department of Transportation website and searching for relevant documents. We do not know of any good way to automate this process. To find these documents, we also manually searched the Internet for the relevant Complete Streets policies already identified by CPHLR, which were typically PDF documents.

4.1.2 Local-Level Statutes and Regulations

Unlike state code and regulations, which can be found for all 50 states on Justia or in a commercial search engine, finding local statues and regulations requires individually searching for each jurisdiction of interest and identifying the service that hosts its code. There is not one commercial search engine that provides access to the laws for a significant majority of local jurisdictions. We found this reflected in the ten cities included in the Local Inclusionary Zoning policy surveillance study. For 9 of the 10 jurisdictions, municipal codes were hosted on 3 different third-party search engines: the code for seven of the jurisdictions was hosted in Municode, one jurisdiction (Burlington, VT) was hosted in Code Publishing, one (Lake Forest, IL) was hosted in American Legal Publishing, and one (San Diego, CA) hosted its municipal code on its own website. For our Local Inclusionary Zoning study consisting of ten cities, it was not problematic to access each jurisdiction individually, but it is a concern for larger local policy surveillance studies.

In contrast to what we found with the search engines hosting state laws, the search engines hosting municipal code make it easy to bulk download a jurisdiction’s entire legal code in a machine-readable format. American Legal Publishing allows one to export the jurisdiction’s entire code as one plain text, PDF, or Microsoft Word document. Code Publishing offers the ability to export the jurisdiction’s entire code in XML, RTF, PDF, or plain text; we were successfully able to download Burlington’s entire code in plain text, but the site timed out when trying to download it in XML. Municode allows one to download a jurisdiction’s entire code as a Microsoft Excel file. This Excel file contains separate columns for the law’s title (typically the section and subsection number), subtitle (the text of the law’s title), and content (the actual text of the law). eCode360, another search engine for local laws, allows users to download a jurisdiction’s entire code in PDF without a subscription and in Microsoft Word with a subscription.

For our experimentation on the Local Inclusionary Zoning dataset, we downloaded the entire municipal code for each of the 9 jurisdictions that are hosted on a third-party search engine. For the seven jurisdictions hosted on Municode, sometimes the text of the section of the law is too large to fit in a Microsoft Excel cell. The Excel file then says “Content is too large for cell” in the content field rather than the actual text of the section. Upon inspection, these laws that were too big typically included multiple large tables. Since the proportion of laws that were too large to fit in an Excel cell is only 1-2% of the law for each jurisdiction hosted on Municode, we omitted these laws from our dataset rather than obtaining their text individually. Despite the low percentage of laws affected, obtaining them would still require individually downloading the text of 21-42 laws for each jurisdiction, which was too time-consuming. We manually verified that none of the relevant inclusionary zoning laws were included in these omitted laws.

The City of San Diego hosts its municipal code as PDF documents on its own website, with one document for each Division of the city code. It is not possible to bulk download the city’s entire code, so we downloaded the PDF for each Division that contains the text “inclusionary zoning” to replicate the search procedure described in CPHLR’s research protocol.

It should be noted that it is not uncommon for jurisdictions to have a zoning code that is separate from their municipal code. This was the case for three of the cities included in the Local Inclusionary Zoning study. Cambridge, MA and Irvine, CA hosted both their municipal code and their zoning code on Municode, so we were able to obtain the entirety of both codes as a Microsoft Excel file. Stamford, CT hosts its municipal code on Municode but hosts its zoning code on its own website as one large PDF file.

4.1.3 Storing Documents in a Machine-Readable Format

As described above, we obtained documents for 50 states and DC and 10 municipal jurisdictions in a variety of formats: Microsoft Word documents from LexisNexis, XML and HTML files from Public.Resource.Org, JSON results from Massachusetts and New York’s APIs, Microsoft Excel files from Municode, plain text downloads from American Legal Publishing and Code Publishing, and PDF documents from individual state and municipality websites. Our goal was to store the plain text of these legal documents in a single database that could be used for further search and analysis. Our database schema included a unique id for each law, the jurisdiction the law was from, the title of the law, and the plain text of the law.

We combined the Complete Streets and Local Inclusionary Zoning data into a single database. Since most policy surveillance work is focused on a single jurisdiction at a time, it is easy to combine all the data into one database and then filter to the jurisdictions of interest. Additionally, using a single database facilitated experimentation in evaluating models trained on Complete Streets Data on Local Inclusionary Zoning data and vice versa.

To get the plain text from these diverse document types, we used various strategies summarized in Table 3. For machine readable data formats, we used the information stored in the document’s tags to identify a law’s title and text. For documents that are not machine readable, we used Apache Tika[[12]](#footnote-13) to extract the text. Apache Tika is an open-source toolkit that extracts text and metadata from over a thousand different filetypes.

Table 3. Extracting Plain Text from Legal Data Sources

|  |  |  |
| --- | --- | --- |
| **Data Source** | **Data Format** | **Processing Method** |
| LexisNexis | Microsoft Word | Apache Tika to extract text; Python code to remove header and footer containing LexisNexis-specific metadata |
| State policy documents | PDF | Apache Tika to extract text; no further processing due to heterogeneity of data |
| Public.Resource.Org State Laws | HTML | Python code using HTML tags to extract law title and text |
| Public.Resource.Org State Regulations | XML | Python code using XML tags to extract law title and text |
| Massachusetts API | JSON | Python code using JSON keys to extract law title and text |
| New York API | JSON | Python code using JSON keys to extract law title and text |
| Municode | Microsoft Excel | Convert to CSV using Microsoft Excel application; select columns containing law title and text |
| American Legal Publishing | Plain text | Python code using spacing and formatting cluse to segment entire code into individual laws |
| Code Publishing | Plain text | Python code using spacing and formatting clues to segment entire code into individual laws |

For much of our future work, especially the document vectors (described in section 4.2.2) and the dense passage retrieval (described in section 4.2.4), an entire law would likely be too long to store in memory while running these algorithms. Therefore, we segmented each law into subsections of up to 200 words without splitting the document in the middle of a sentence. We used the Haystack[[13]](#footnote-14) library, an open-source Python package for document ranking and question-answering, to perform this segmentation.

Segmenting the laws into 200-word sections leaves open the possibility that an important section of the document will be separated into different subsections. In this work, we manually verified that the answers to all of the policy surveillance questions were entirely contained within a subsection. We only found one instance where the answer to a policy surveillance question was split into two different subsections, and we manually corrected those subsections so that the policy surveillance answer was contained in a single section. In future work using machine learning to conduct policy surveillance, we would not want to include a manual step verifying that the correct answers are not split among subsections. To avoid this, one could use a sliding window to create the 200-word sections, where adjacent sections would have a certain number of terms in common (i.e., the first section would contain words 0-200, the second section would contain words 150-350, the third section would contain 300-500, etc.), thereby ensuring that an answer would be fully contained within at least one section.

4.2 How Do We Find Relevant Documents in a Legal Corpus with Natural Language Processing (NLP)?

Now that we have accumulated a large collection of legal documents, we must identify those that are relevant to the topic at hand. In the manual policy surveillance process, researchers typically devise a set of search terms based on the background research they have done on the topic. Once researchers have identified a set of key words, the results of the searches performed with these keywords will often reveal additional useful search terms or identify ways in which the initial search terms need to be modified. Researchers also typically peruse tables of contents to find additional relevant documents that were not identified with their search queries.

We applied several strategies to aid in both the query development and document retrieval processes, reducing the amount of time researchers must spend devising search queries and looking for documents the queries may have missed:

* In section 4.2.1, we describe training and evaluating word embeddings, which can help human researchers augment their search queries.
* We describe training and evaluating document embeddings (section 4.2.2) and training a topic model over the set of documents (section 4.2.3), both of which provide a method for searching based on finding similar documents to an exemplar document.
* In section 4.2.4, we discuss several experiments in document retrieval, which involve identifying documents which are similar to a specific question, directly using the policy surveillance questions in the search process.

4.2.1 Using Word Embeddings to Enhance Search Query Generation

Word embeddings[[14]](#footnote-15) use a large corpus of documents to form a vector representation of each word found in the document based on how those terms appear in the text. Words that appear in similar contexts will be mapped to a similar location in the vector space, so that (for example), the terms “big” and “large” will have similar vector representations. See Figure 3 for an example of words mapped into a two-dimensional vector space, where words with similar meanings are located near one another.

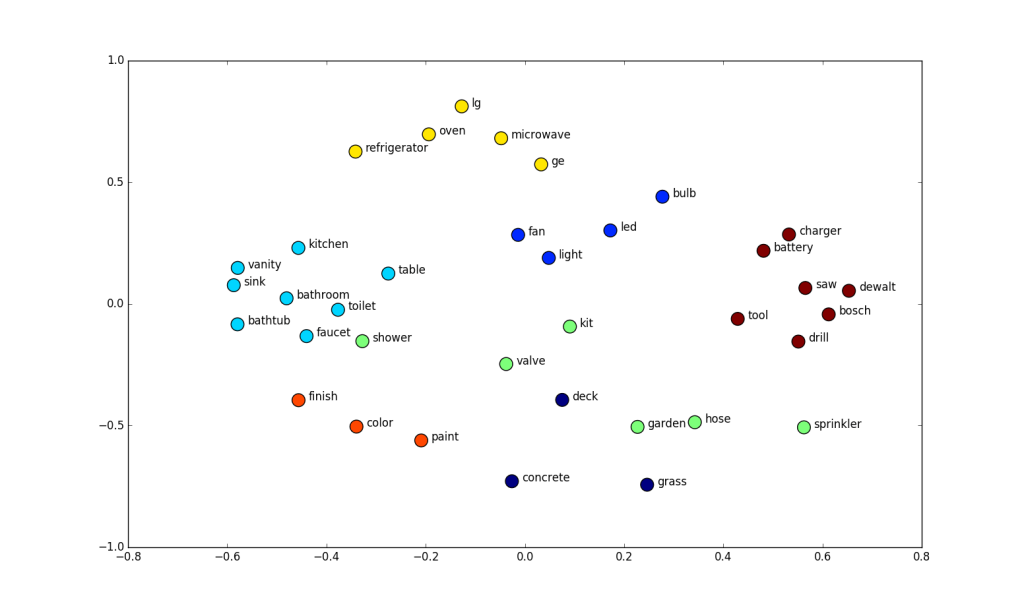


Figure 3. Visualization of Two-Dimensional Vector Representations of Words. Image from Neptune.ai.

Existing general-purpose word embeddings are trained on the text of Wikipedia. There are also word embeddings trained specifically on legal text: SigmaLaw[[15]](#footnote-16), trained on over 39,000 US legal cases; Law2Vec[[16]](#footnote-17), trained on over 123,000 legal documents in English from around the world; and legal embeddings from a prior MITRE research effort, trained on the United States Code, the Code of Federal Regulations, and the Internal Revenue Service policies and procedures. The quality of a set of word embeddings is difficult to evaluate intrinsically and requires a human to judge their usefulness. The first four columns of Table 4 show some example results from these pretrained word embeddings. The first column contains an initial query term relevant to the Complete Streets or Local Inclusionary zoning policy surveillance studies, and the results under each word embedding show the most similar words to that query term.

Table 4. Sample Results from Various Word Embeddings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Initial Query:** | **SigmaLaw** | **Law2Vec** | **MITRE – Prior Effort** | **MITRE – This Effort** |
| **street** | boulevard, corner, avenue, road, streets | avenue, lane, terrace, road, cheapside | nw, avenue, st., n.w., suite | thoroughfare, streets, parkway, alley, boulevard |
| **transit** | transp, metro, airports, metropolitan, transportation | port, carnet, transiting, carnets, gddtyo | transiting, transportation, transported, intermodal, destination | mass transit, transportation, mass transportation, passenger rail, light rail |
| **pedestrian** | pedestrians, roadways, crosswalk, vehicular, walkways | vehicular, pedestrians, cycleways, crossing, underpass | vehicular, bicycle, walkways, roadways, roadway | pedestrians, bicycle, pedestrian traffic, bicyclists, pedestrian circulation |
| **zoning** | rezoning, ordinances, rezoned, ordinance, bza | airport, igloolik, arviat, flon, kindersley | ordinances, ordinance, bylaw, redevelopment, seashore | zoning ordinances, local zoning, land-use, ordinances, zoning ordinance |
| **affordable** | affordability, subsidized, housing, residential, decent | affordability, low-income, supportive, decent, moderate-income | affordability, aca, ppaca, decent, care | affordable housing, affordability, cost-effective, low-income, low income |
| **housing** | hous, fha, multifamily, hud, revitalization | urban, regeneration, brockton, leasehold, redevelopment | multifamily, dwellings, homeownership, dwelling, rental | rental housing, affordable housing, multi-family housing, congregate housing, housing developments |
| **complete streets** | -- | -- | -- | geometric design, multimodal transportation, multi-modal, context sensitive |

We can see that some of the Law2Vec results are strange, perhaps because they include documents from other countries. The MITRE embeddings trained on the US Code sometimes better reflect federal policy than state and local policy (i.e., “affordable” refers to the Affordable Care Act rather than affordable housing). Therefore, we also trained word embeddings on all the state and local codes and regulations we collected as described in Section 4.1. The final column of Table 4 shows results from the embeddings we trained on this state and local legal text. We can see that for all sample query terms, the most similar terms returned are highly relevant to the initial query. Unlike most open-source embeddings, the training procedure we employed in this effort includes multi-word phrases in the word embeddings. This means that we can query for a whole phrase such as “complete streets” and get results, which is not possible with any of the other word embeddings, as shown in the last row of Table 4. This is also why some of the most similar terms these word embeddings returned for the single word queries in rows 1-6 of Table 4 are phrases rather than single words, which we do not see with the other word embeddings.

**Recommendation**: We recommend making the word embeddings trained for this effort available to policy surveillance practitioners. To apply word embeddings in the policy surveillance process, once researchers have identified a few search terms they feel are useful for their task, they can use the word embeddings to identify additional terms which are located near their original search terms in the vector space. This way they will be able to quickly identify additional relevant search terms without needing to do further research.

4.2.2 Document Vectors as a Document Search Strategy

The word embeddings described in the previous section have been extended to apply to sentences and paragraphs in a technique known as Doc2Vec[[17]](#footnote-18). Rather than mapping individual terms in a vector space, Doc2Vec seeks to map longer texts, such as sentences, paragraphs, or entire short documents, into a vector space. Just as word embeddings map words that appear in similar contexts to nearby locations in the vector space, documents that use similar terms in similar ways will also be mapped to nearby locations in the vector space. Given a relevant document (perhaps identified by standard keyword search techniques), we expect that additional relevant documents would be found nearby in the document vector embedding space. This technique would also promote generalization across multiple jurisdictions, as one could use a relevant document for one jurisdiction to find similar documents in other jurisdictions.

We trained a document vector model on all the legal documents we collected as described in Section 4.1, using the 200-word sections described in Section 4.1.3. Since we are ultimately interested in identifying the specific sentence or sentences that provide the answer to a research question, using these shorter documents will not be problematic; we do not necessarily need to retrieve the entire text of a longer law if only a small portion is of interest.

To use a document vector model, one starts with a document of interest, and uses it to find other documents with similar content. The document vector model was very successful at finding similar documents. Table 5 shows some sample results from the document vector model. (The section number indicates the specific 200-word section identified. If no section is listed, the document was short enough to fit in one section.)

Table 5. Sample Results from Document Vector Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Initial Document** | **Most Similar Documents** | | | |
| West Virginia, Chapter 17 Roads and Highways – Article 4A Complete Streets Act – Section 17-4A-1 Complete Streets (section 1) | Connecticut Department of Transportation Policy No. EX.0. – 31. Complete Streets (section 1) | Vermont, Agency 14 Agency of Transportation – Sub-Agency 010 Transportation Board – Chapter 019 Vermont State Standards for the Design of Transportation – (section 3) | Rhode Island, Title 24 Highways – Chapter 16 Safe Access to Public Roads – Section 24-16-1 Legislative findings | Virginia, Virginia Department of Transportation Policy for Integrating Bicycle and Pedestrian Accommodations (section 5) |
| Delaware Department of Transportation Complete Streets Policy (Executive Order 6) (section 5) | Delaware Department of Transportation Complete Streets Policy (Executive Order 6) (section 6) | Maine Department of Transportation Complete Streets Policy (section 9) | VDOT Departmental Memorandum 2-12, Implementation of the CTB Policy for Integrating Bicycle and Pedestrian Accommodations (section 11) | North Carolina Department of Transportation Complete Streets Implementation Guide (section 13) |
| Evanston, Housing Regulations – Inclusionary Housing 5-7-4. Requirements | Boulder, Land Use Code – Building Types M-1-20. Measurement of Building Type Requirements (section 1) | Lake Forest, Chapter 158 Inclusionary Housing 158.04 Percentage of Affordable Housing Units Required | Oregon, Land Conservation and Development Department – 660-038-0040 Determine the Mix of Dwelling Units Needed (section 2) | San Diego, Chapter 14 Article 3 Division 10 43.1002 Application of Complete Communities Housing Solutions Regulations |
| Stamford Zoning Code (section 11) | Stamford Zoning Code (section 12) | Massachusetts, Department of Housing and Community Development – 760 59-02 Definitions | San Diego, Chapter 14 Article 2 Division 6 42.0640 Development Impact Fees for Public Facilities and Spaces | Boulder, Land Use Code – Inclusionary Housing 9-13-10. Options for Satisfaction of Inclusionary Housing Requirement |

We can see from the titles that the document vector model successfully identified documents similar in content to the initial seed document. We can see that this model can identify similar text discussing very specific subjects. For example, the Evanston Inclusionary Housing Requirements law listed in the third line of Table 5 discusses the percentage of housing units in a development that must be set aside for affordable housing, which is precisely the aspect of affordable housing discussed in the identified Lake Forest and San Diego laws. In this example, the state-level law from Oregon identified discusses the percentages of different types of density required, so it is also similar in content even though it is not from the same type of jurisdiction.

**Recommendation**: We recommend pursuing document search using Doc2Vec in future policy surveillance work. We would hope to train a document vector model using more state laws obtained from Justia, experiment with slight variations in the Doc2Vec training parameters to identify the best document vector model and design a user interface that would allow a user to choose a document of interest and show them similar documents in the other jurisdictions they wish to search through.

4.2.3 Topic Modeling as a Document Search Strategy

A topic model trained over a set of documents identifies groups of words that meaningfully cooccur in those documents. Each group of words is said to form a topic, and one can define a distribution over the topics in the model for each document in the corpus. Therefore, one can use a topic model to define a representation for each document in a latent vector space. Documents that discuss similar topics will then be close to one another in this vector space. This raises the possibility of using the topic model representations of each document as the basis for document search. Given a relevant document (perhaps identified by standard keyword search techniques), we expect that additional relevant documents discuss similar topics and would be found nearby in the topic model vector space. This technique would also promote generalization across multiple jurisdictions, as one could use a relevant document for one jurisdiction to find similar documents in other jurisdictions.

We trained topic models with 20, 40, and 80 topics on all the legal documents we collected as described in Section 4.1. We chose to train topic models with a range of sizes since it can be difficult to know a priori how many topics are adequate for a given corpus. A model with more topics can identify topics that are discussed less frequently in a corpus or divide a general topic into two more specific topics, but a model with too many topics runs the risk of generating redundant topics. To maintain consistency, the documents we used were segmented into the same 200-word sections described in Section 4.1.3. We used the Topic Modeling Neural Toolkit (TMNT)[[18]](#footnote-19), a MITRE-developed open-source toolkit that computes topic models using a neural network variational autoencoder[[19]](#footnote-20). We found that the resulting topic models showed very clear topics capturing the large variety of subjects discussed in state and local laws. Table 6 shows the terms most significantly associated with selected topics from a 40-topic model.

Table 6. Selected Topics from a 40-Topic Model

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| topic 1 | employer | employers | unemployment | claimant | apprenticeship |
| topic 2 | foster | adoptive | child | children | parent |
| topic 4 | appoint | appointed | senate | vice | elected |
| topic 6 | editorial | current | amending | government | penal |
| topic 7 | evacuation | alarm | staff | volunteers | orientation |
| topic 8 | winning | wagers | play | betting | wager |
| topic 9 | minor | permit | modification | project | major |
| **topic 10** | **ordinance** | **zoning** | **lot** | **dwelling** | **voters** |
| topic 11 | credit | semester | union | courses | clock |
| topic 13 | lands | landowner | vegetation | scenic | trees |
| topic 14 | moneys | proceeds | deed | money | lien |
| topic 16 | taxable | income | taxpayer | deduction | taxation |
| topic 17 | disabilities | aging | developmental | blind | respite |
| topic 19 | meeting | informal | quorum | meetings | chairman |
| topic 20 | customer | customers | pipeline | cable | gas |
| topic 22 | correctional | felony | defendant | custody | inmate |
| topic 23 | dollars | penalty | violates | violation | misdemeanor |
| topic 24 | dispute | customer | appellant | billing | overpayment |
| topic 25 | worth | surplus | insurer | dealer | securities |
| topic 26 | injection | practitioner | prescribing | physician | pain |
| topic 27 | ambulatory | inpatient | hospital | outpatient | diem |
| topic 29 | brand | eggs | agriculture | seed | milk |
| **topic 30** | **vehicle** | **vehicles** | **driver** | **passenger** | **truck** |
| topic 33 | remediation | contaminated | remedial | constituents | groundwater |
| topic 34 | deposition | testimony | pleadings | prehearing | witnesses |
| topic 37 | student | school | schools | charter | tuition |
| topic 39 | address | mailing | registration | number | numbers |

In this model, documents with high scores in Topic 30 would be relevant to the Complete Streets policy surveillance study, while documents with high scores in Topic 10 would be relevant to the Local Inclusionary zoning study.

Table 7 shows topics from a 20-topic model. We can see that the 40-topic model provides more granularity than a 20-topic model. This 20-topic model does not have any topics directly related to Complete Streets or Local Inclusionary Zoning. We can also see that in the 20-topic model, there is one topic (topic 8) related to doctors and hospitals, while in the 40-topic model, that topic has been split into two finer-grained topics (topics 26 and 27), one more focused on doctors and one more focused on hospitals.

Table 7. Topics from a 20-Topic Model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| topic 0 | radioactive | phosphorus | soil | solids | ml | chloride |
| topic 1 | instructor | appraiser | licensure | accredited | passing | accreditation |
| topic 2 | obligor | overpayment | debtor | tribunal | father | appellant |
| topic 3 | insurer | insurers | premium | insurance | filings | medicare |
| topic 4 | violates | violating | guilty | fine | punished | suspension |
| topic 5 | electricity | fuel | diesel | voc | steam | gasoline |
| topic 6 | poultry | utensils | meat | communicable | establishments | cooking |
| topic 7 | voter | ticket | tag | transporter | shipment | tickets |
| topic 8 | therapy | therapist | nurse | hospice | medication | medications |
| topic 9 | senate | compact | governor | appointed | retirement | interstate |
| topic 10 | broker | trustee | adviser | merger | guarantor | securities |
| topic 11 | iep | lea | childhood | nonpublic | kindergarten | elementary |
| topic 12 | project | projects | grant | funding | grants | grantee |
| topic 13 | levied | levy | roll | taxes | redemption | bonds |
| topic 14 | ideas | solve | dok | interpret | analyze | equations |
| topic 15 | calibration | surveillance | audits | audit | exposure | infection |
| topic 16 | tanf | annuity | sick | leave | contributions | spouse |
| topic 17 | presiding | argument | quorum | motions | prehearing | chair |
| topic 18 | constituents | nox | landfill | env | leachate | effluent |
| topic 19 | egress | exit | min | doors | door | sprinkler |

Table 8 shows selected topics from an 80-topic model. This model includes topics relevant to the policy surveillance studies of interest (topics 55 and 59), but we can also see that some of the topics start to get redundant. For example, topics 50 and 79 both appear to be about education, and it is hard to discern a distinction between the two.

Table 8. Selected Topics from an 80-Topic Model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| topic 49 | hospital | acute | hospitals | patients | care | patient |
| topic 50 | students | student | college | teacher | academic | patient |
| topic 51 | application | eligible | eligibility | applicants | ineligible | verify |
| topic 52 | officer | director | chief | basic | executive | duties |
| topic 53 | corporate | foreign | applicant | partner | existence | background |
| topic 54 | insurer | pharmacist | emission | presiding | pharmacy | emissions |
| **topic 55** | **transport** | **transportation** | **highways** | **vehicle** | **alcoholic** | **permits** |
| topic 56 | waste | tank | fuel | hazardous | holder | wastes |
| **topic 59** | **comprehensive** | **boundaries** | **county** | **ordinance** | **counties** | **urban** |
| topic 60 | central | data | criminal | collection | registry | judicial |
| topic 61 | assets | financial | asset | underground | liabilities | bank |
| topic 63 | children | good | age | child | body | covering |
| topic 64 | applicant | proposed | factors | sought | costs | detailed |
| topic 65 | expenses | allowable | reimbursement | costs | devices | excess |
| topic 67 | facility | transfers | transfer | resident | transferred | legal |
| topic 68 | students | prescription | learning | cleaning | pharmacist | abuse |
| topic 69 | retention | disposal | drug | administration | controlled | series |
| topic 71 | list | offices | places | large | selected | select |
| topic 76 | division | assembly | general | assist | promote | directed |
| topic 77 | child | milk | casing | client | pharmacist | valve |
| topic 78 | managed | stock | controlled | species | class | plants |
| topic 79 | school | district | college | students | schools | academic |

Due to current time constraints, we were unable to implement a system that used each document’s topic vector representation to search for similar documents.

**Recommendation**: Given the observed quality of the topic models, we would recommend pursuing document search using topic models in future policy surveillance work.

We would train topic models using more state laws obtained from Justia. A 20-topic model is probably too small for this amount of data, but if we do include all the state laws from Justia, we might find that the 80 topic models are not as redundant as we found in this experimentation.

**Recommendation**: We recommend training 40 and 80 topic models on all of the legal documents we are able to obtain and designing a user interface that would allow a user to choose a document of interest and show them documents with similar topic vector representations in the other jurisdictions they wish to search through.

4.2.4 Document Retrieval in Response to a Question

The preceding two sections examined ways to improve document retrieval based on finding search terms similar to a given search term or additional documents similar to an exemplar document. However, in the policy surveillance process, researchers have defined specific questions they want answered. Many information retrieval techniques have been established to identify a document that is relevant to a given question. In section 4.2.4.1, we explore using bag-of-words methods to identify the documents that contain the answer to a question of interest. In section 4.2.4.2, we experiment with dense passage retrieval for document retrieval.

4.2.4.2 Bag-of-Words Methods for Document Retrieval

We used Term Frequency-Inverse Document Frequency (TF-IDF) and Best Match 25 (BM-25), to identify the laws containing the answer to a question of interest. TF-IDF and BM-25 are bag-of-words methods that are based on calculating term frequencies over the entire document corpus. They do not require any training to implement. TF-IDF[[20]](#footnote-21) measures how frequently the query terms appear in a document of interest when compared to how frequently the query terms appear in the whole document corpus. The idea is that a document that contains many mentions of a term in a query, especially when those terms in the query do not appear in the whole corpus very often, is highly relevant to that query. BM-25[[21]](#footnote-22) is a modification of TF-IDF that accounts for the document length and repeated mentions of the query terms. Results for these methods of document retrieval on the test sets for the Complete Streets dataset are shown in Table 9. Results for these methods of document retrieval on the test sets for the Local Inclusionary Zoning dataset are shown in Table 10. In this set of experiments, we used the questions as they were written in the initial policy surveillance study, without including the jurisdiction in the question text. In the tables, K represents the number of documents retrieved, i.e., retrieving the highest scoring document, the 5 highest scoring documents, etc. Accuracy is the percentage of questions for which the correct document was retrieved. MRR (mean reciprocal rank) is a metric used in information retrieval that calculates the average rank of the desired search result in an ordered list of search results, more heavily rewarding results that are ranked higher in the search results.

Table 9. Bag of Words Document Retrieval Results for the Complete Streets Test Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **K** | **Accuracy** | **MRR** |
| TF-IDF | 1 | 0.139 | 0.139 |
| TF-IDF | 5 | 0.221 | 0.170 |
| TF-IDF | 10 | 0.329 | 0.184 |
| TF-IDF | 20 | 0.387 | 0.188 |
| TF-IDF | 100 | 0.559 | 0.192 |
| BM-25 | 1 | 0.296 | 0.174 |
| BM-25 | 5 | 0.450 | 0.244 |
| BM-25 | 10 | 0.541 | 0.255 |
| BM-25 | 20 | 0.607 | 0.260 |
| BM-25 | 100 | 0.674 | 0.266 |

Table 10. Bag of Words Document Retrieval Results for the Local Inclusionary Zoning Test Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **K** | **Accuracy** | **MRR** |
| TF-IDF | 1 | 0.141 | 0.141 |
| TF-IDF | 5 | 0.253 | 0.177 |
| TF-IDF | 10 | 0.396 | 0.196 |
| TF-IDF | 20 | 0.493 | 0.203 |
| TF-IDF | 100 | 0.722 | 0.209 |
| BM-25 | 1 | 0.174 | 0.174 |
| BM-25 | 5 | 0.389 | 0.244 |
| BM-25 | 10 | 0.470 | 0.255 |
| BM-25 | 20 | 0.548 | 0.260 |
| BM-25 | 100 | 0.744 | 0.266 |

We can see that at higher values of K the bag-of-words methods do well at identifying the document containing the answer to a policy surveillance question. For example, BM-25 achieves 0.674 accuracy on the Complete Streets data and 0.744 accuracy on the Local Inclusionary Zoning data at K=100. TF-IDF is only slightly worse, with accuracy of 0.559 on Complete Streets and 0.722 on Local Inclusionary Zoning at K=100. But it is unlikely that policy surveillance researchers would want to look through 100 documents to find the one containing the relevant information, and the accuracy at smaller values of K (0.45 at K=5 for Complete Streets data) is probably not good enough to be useful to a policy surveillance researcher.

4.2.4.2 Dense Passage Retrieval for Document Retrieval

Dense passage retrieval[[22]](#footnote-23) uses a neural network based on Bidirectional Encoder Representations from Transformers (BERT)[[23]](#footnote-24) to jointly train vector embeddings for both the passages in the document corpus and a training set of questions. (BERT is a neural language model trained to predict missing words and sentences in text. In learning to perform these tasks, BERT encodes a large amount of linguistic and real-world knowledge, making it an ideal starting point for a wide range of natural language processing tasks.) At search time, the dense passage retriever encodes a vector representation for a given question and identifies the passages whose vector representations are most similar to that question’s vector representation. For the dense passage retrieval, we compared an open-source implementation based on the Natural Questions[[24]](#footnote-25) dataset (derived from Wikipedia) to retrievers trained directly on the questions and passages from the policy surveillance datasets and the open-source pre-trained retriever further fine-tuned on the policy surveillance datasets. For training and fine-tuning the dense passage retrievers, we used the data splits described in Section 3.3, using the training split to train the model for 40 epochs, or passes through the data, and the validation split to choose which model to evaluate on the testing split. We used the Haystack library described in Section 4.1.3 to perform all experiments. To maintain consistency, the documents we searched over were the same legal documents segmented into 200-word sections described in Section 4.1.3.

Results for dense passage retrieval on the test sets for the Complete Streets dataset are shown in Table 11. Results for dense passage retrieval on the test sets for the Local Inclusionary Zoning dataset are shown in Table 12. In this set of experiments, we used the questions as they were written in the policy surveillance study, without including the jurisdiction in the question text. In the tables, K represents the number of documents retrieved, i.e., retrieving the highest scoring document, the 5 highest scoring documents, etc. Accuracy is the percentage of questions for which the correct document was retrieved. MRR (mean reciprocal rank) is a metric used in information retrieval that calculates the average rank of the desired search result in an ordered list of search results; more heavily rewarding results that are ranked higher in the search results.

Table 11. Dense Passage Retrieval Results for the Complete Streets Test Dataset

| **Method** | **Training Dataset** | **K** | **Accuracy** | **MRR** |
| --- | --- | --- | --- | --- |
| Open-Source DPR |  | 1 | 0.039 | 0.039 |
| Open-Source DPR |  | 5 | 0.085 | 0.057 |
| Open-Source DPR |  | 10 | 0.139 | 0.065 |
| Open-Source DPR |  | 20 | 0.205 | 0.069 |
| Open-Source DPR |  | 100 | 0.381 | 0.074 |
| Trained DPR | Small Dataset (5 states) | 1 | 0.003 | 0.003 |
| Trained DPR | Small Dataset (5 states) | 5 | 0.012 | 0.005 |
| Trained DPR | Small Dataset (5 states) | 10 | 0.030 | 0.008 |
| Trained DPR | Small Dataset (5 states) | 20 | 0.073 | 0.011 |
| Trained DPR | Small Dataset (5 states) | 100 | 0.184 | 0.013 |
| Trained DPR | Large Dataset (10 states) | 1 | 0.003 | 0.003 |
| Trained DPR | Large Dataset (10 states) | 5 | 0.030 | 0.014 |
| Trained DPR | Large Dataset (10 states) | 10 | 0.042 | 0.015 |
| Trained DPR | Large Dataset (10 states) | 20 | 0.070 | 0.017 |
| Trained DPR | Large Dataset (10 states) | 100 | 0.205 | 0.019 |
| Finetuned DPR | Small Dataset (5 states) | 1 | 0.021 | 0.021 |
| Finetuned DPR | Small Dataset (5 states) | 5 | 0.060 | 0.036 |
| Finetuned DPR | Small Dataset (5 states) | 10 | 0.082 | 0.039 |
| Finetuned DPR | Small Dataset (5 states) | 20 | 0.142 | 0.043 |
| Finetuned DPR | Small Dataset (5 states) | 100 | 0.354 | 0.048 |
| Finetuned DPR | Large Dataset (10 states) | 1 | 0.045 | 0.045 |
| Finetuned DPR | Large Dataset (10 states) | 5 | 0.106 | 0.068 |
| Finetuned DPR | Large Dataset (10 states) | 10 | 0.169 | 0.076 |
| Finetuned DPR | Large Dataset (10 states) | 20 | 0.290 | 0.084 |
| Finetuned DPR | Large Dataset (10 states) | 100 | 0.520 | 0.089 |

Table 12. Dense Passage Retrieval Results for the Local Inclusionary Zoning Test Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **K** | **Accuracy** | **MRR** |
| Open-Source DPR | 1 | 0.059 | 0.059 |
| Open-Source DPR | 5 | 0.207 | 0.110 |
| Open-Source DPR | 10 | 0.263 | 0.117 |
| Open-Source DPR | 20 | 0.289 | 0.119 |
| Open-Source DPR | 100 | 0.519 | 0.124 |
| Trained DPR | 1 | 0.000 | 0.000 |
| Trained DPR | 5 | 0.059 | 0.018 |
| Trained DPR | 10 | 0.063 | 0.018 |
| Trained DPR | 20 | 0.070 | 0.019 |
| Trained DPR | 100 | 0.230 | 0.022 |
| Finetuned DPR | 1 | 0.092 | 0.093 |
| Finetuned DPR | 5 | 0.233 | 0.140 |
| Finetuned DPR | 10 | 0.289 | 0.146 |
| Finetuned DPR | 20 | 0.359 | 0.151 |
| Finetuned DPR | 100 | 0.704 | 0.159 |

The open-source dense passage retriever does not work as well as the bag-of-words methods (accuracy of 0.381 on Complete Streets and 0.519 on Local Inclusionary Zoning at K=100 compared to BM-25’s accuracy of 0.644 for Complete Streets and 0.744 for Local Inclusionary Zoning), likely because the language used in asking factual questions about information in Wikipedia articles is very different from the language found in legal text. The patterns for smaller values of K are similar. (For comparison purposes, the open-source dense passage retrieval scores 0.794 accuracy at K=20 and 0.860 accuracy at K=100 on the Natural Questions benchmarking dataset. Results for smaller values of K were not reported.)

The results from training a dense passage retriever from scratch were quite poor, although we can see in Table 11 that we did see improved performance when training on the questions from ten states in the Complete Streets dataset when compared with training on the questions from five states, achieving accuracy of 0.520 using 10 states for training compared to accuracy of 0.354 with five states at K=100. This indicates that the data from five or ten jurisdictions is simply not enough to adequately train a dense passage retriever. The results from finetuning the open-source dense passage retrieval show improvement over both the open-source retriever and training a retriever from scratch: for example, the finetuned retriever achieves accuracy of 0.704 at K=100 on the Local Inclusionary Zoning dataset compared to the open-source retriever’s accuracy of 0.519 and the trained retriever’s accuracy of 0.230. However, the TF-IDF and BM-25 methods still outperform the finetuned retrievers across all values of K. In general, we see slightly higher performance on the models trained on the Local Inclusionary Zoning dataset than the models trained on the Complete Streets dataset. This is likely because the municipalities in the Local Inclusionary Zoning study have smaller bodies of law than the states in the Complete Streets study, so it is inherently somewhat easier to find the correct legal documents.

One hypothesis we had about the poor performance of the trained dense passage retrievers was that the training procedure was compromised by including the same question multiple times. For example, the training data would contain the question “Is the policy mandatory for developers?” multiple times, and in one instance the correct passage would be the passage from Highland Park, and in another instance the correct passage would be the passage from Santa Fe. Having multiple possible answers for an identical question might cause problems for the dense passage retriever, as in the training procedure answers to other questions are chosen as negative examples for the system to learn from. It is possible that the training procedure could choose the Santa Fe passage as a negative example for the Highland Park question, although that passage is also a correct answer to the question.

To verify if this was the case, we also evaluated the dense passage retrieval methods using the version of the question that included the jurisdiction. Results of all methods of document retrieval on the test sets for the Complete Streets dataset using this question formulation are shown in Table 13. Results of all methods of document retrieval on the test sets for the Local Inclusionary Zoning dataset using this question formulation are shown in Table 14.

Table 13. Dense Passage Retrieval Results for the Complete Streets Test Dataset, Including the Jurisdiction in the Question Text

| **Method** | **Dataset** | **K** | **Accuracy** | **MRR** |
| --- | --- | --- | --- | --- |
| Open-Source DPR |  | 1 | 0.039 | 0.039 |
| Open-Source DPR |  | 5 | 0.082 | 0.054 |
| Open-Source DPR |  | 10 | 0.112 | 0.058 |
| Open-Source DPR |  | 20 | 0.154 | 0.061 |
| Open-Source DPR |  | 100 | 0.290 | 0.064 |
| Trained DPR | Small Dataset (5 states) | 1 | 0.002 | 0.015 |
| Trained DPR | Small Dataset (5 states) | 5 | 0.063 | 0.034 |
| Trained DPR | Small Dataset (5 states) | 10 | 0.097 | 0.038 |
| Trained DPR | Small Dataset (5 states) | 20 | 0.127 | 0.040 |
| Trained DPR | Small Dataset (5 states) | 100 | 0.221 | 0.042 |
| Trained DPR | Large Dataset (10 states) | 1 | 0.030 | 0.030 |
| Trained DPR | Large Dataset (10 states) | 5 | 0.124 | 0.068 |
| Trained DPR | Large Dataset (10 states) | 10 | 0.160 | 0.073 |
| Trained DPR | Large Dataset (10 states) | 20 | 0.224 | 0.077 |
| Trained DPR | Large Dataset (10 states) | 100 | 0.357 | 0.080 |
| Finetuned DPR | Small Dataset (5 states) | 1 | 0.006 | 0.006 |
| Finetuned DPR | Small Dataset (5 states) | 5 | 0.042 | 0.019 |
| Finetuned DPR | Small Dataset (5 states) | 10 | 0.067 | 0.023 |
| Finetuned DPR | Small Dataset (5 states) | 20 | 0.103 | 0.025 |
| Finetuned DPR | Small Dataset (5 states) | 100 | 0.248 | 0.029 |
| Finetuned DPR | Large Dataset (10 states) | 1 | 0.006 | 0.006 |
| Finetuned DPR | Large Dataset (10 states) | 5 | 0.079 | 0.031 |
| Finetuned DPR | Large Dataset (10 states) | 10 | 0.136 | 0.038 |
| Finetuned DPR | Large Dataset (10 states) | 20 | 0.184 | 0.042 |
| Finetuned DPR | Large Dataset (10 states) | 100 | 0.311 | 0.045 |

Table 14. Document Retrieval Results for the Local Inclusionary Zoning Test Dataset, Including the Jurisdiction in the Question Text

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **K** | **Accuracy** | **MRR** |
| Open-Source DPR | 1 | 0.030 | 0.030 |
| Open-Source DPR | 5 | 0.089 | 0.046 |
| Open-Source DPR | 10 | 0.130 | 0.051 |
| Open-Source DPR | 20 | 0.222 | 0.058 |
| Open-Source DPR | 100 | 0.430 | 0.063 |
| Trained DPR | 1 | 0.000 | 0.000 |
| Trained DPR | 5 | 0.000 | 0.000 |
| Trained DPR | 10 | 0.004 | 0.000 |
| Trained DPR | 20 | 0.004 | 0.000 |
| Trained DPR | 100 | 0.085 | 0.002 |
| Finetuned DPR | 1 | 0.056 | 0.056 |
| Finetuned DPR | 5 | 0.182 | 0.094 |
| Finetuned DPR | 10 | 0.248 | 0.103 |
| Finetuned DPR | 20 | 0.315 | 0.108 |
| Finetuned DPR | 100 | 0.570 | 0.113 |

It turns out that including the jurisdiction in the question text resulted in slightly worse performance across the board. The potential use of an identical question’s passage as a negative example was not problematic; in fact, including the jurisdiction somewhat hindered the model, perhaps because the model was trying to incorporate the jurisdiction’s name in the document retrieval process, even though a jurisdiction’s name is typically not included in the legal text.

Even in the best case, the BM-25 method achieved 54% accuracy with 10 documents for the Complete Streets dataset and 47% accuracy with 10 documents for the Local Inclusionary Zoning dataset. 50% accuracy is probably not high enough to satisfy a policy surveillance practitioner. Accuracy went up when considering 100 documents, but policy surveillance practitioners are unlikely to want to search through 100 documents before they arrive at the one with the correct answer. In contrast, when using the document vector model to search for documents, the top documents retrieved were all relevant.

**Recommendation**: We do not recommend pursuing any of the document retrieval methods we tested based on the question text in future policy surveillance work and instead recommend focusing on using document vector models and topic models to identify documents of interest.

4.3 How Do We Identify Passages from Legal Documents That Answer Questions of Interest?

Given the passages retrieved as relevant to a question, question-answering systems seek to use a subset of that passage to provide an answer to the question. Extractive question-answering systems identify a span, or section, of text from the passages that best answers the question. State of the art question-answering systems start with BERT as the basis for identifying the starting and ending positions of a span of text and are further trained on a question-answering dataset such as the Stanford Question Answering Dataset (SQuAD),[[25]](#footnote-26) another dataset based on Wikipedia.

We compared an open-source question-answering system trained on the SQuAD dataset to a question-answering system trained directly on the questions and passages from the policy surveillance datasets, as well as the open-source question-answering system further fine-tuned on the policy surveillance datasets. For training and fine-tuning the question-answering systems, we used the data splits described in Section 3.3. We used the Haystack library to perform all experiments.

Using a question-answering system requires first identifying relevant passages to extract the answer to a question. Based on the results described in Section 4.2.4, we used BM-25 as our document retrieval method. We first tested the open-source question-answering system trained on the SQuAD dataset with a BM-25 retriever returning both 20 and 100 documents. The results are shown in Table 15 for the test sets for both the Complete Streets and Local Inclusionary Zoning datasets. Since we previously found that including the jurisdiction in the question harmed retrieval performance, we used the questions as they were written in the initial policy surveillance study, without including the jurisdiction in the question text. In Table 15, K and MRR have the same meanings as described in Section 4.2.4. To determine the accuracy of the question-answering system, we deemed a predicted answer to be correct if it was an excerpt of the correct answer text. For example, for the question “Does the policy require inclusion of all ages?”, the supporting answer text for the state of Connecticut was “It is the policy of the department to consider the needs of all users of all abilities and ages (specifically including pedestrians, bicyclists, transit users, and vehicle operators) in the planning, programming, design, construction, retrofit and maintenance activities related to all roads and streets as a means of providing a “safe, efficient transportation network which enhances quality of life and economic vitality”.” The open-source question answering system predicted “It is the policy of the department to consider the needs of all users of all abilities and ages” as its second-ranked answer. We considered this answer to be correct.

Table 15. Open-Source Question-Answering Results for Both Policy Surveillance Datasets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **Number of Documents Retrieved** | **K** | **Accuracy** | **MRR** |
| Complete Streets | 20 | 1 | 0.018 | 0.018 |
| Complete Streets | 20 | 5 | 0.097 | 0.045 |
| Complete Streets | 20 | 10 | 0.160 | 0.054 |
| Complete Streets | 20 | 20 | 0.218 | 0.058 |
| Zoning | 20 | 1 | 0.056 | 0.056 |
| Zoning | 20 | 5 | 0.189 | 0.103 |
| Zoning | 20 | 10 | 0.248 | 0.110 |
| Zoning | 20 | 20 | 0.352 | 0.117 |
| Complete Streets | 100 | 1 | 0.003 | 0.003 |
| Complete Streets | 100 | 5 | 0.036 | 0.015 |
| Complete Streets | 100 | 10 | 0.085 | 0.021 |
| Complete Streets | 100 | 20 | 0.145 | 0.025 |
| Complete Streets | 100 | 100 | 0.269 | 0.028 |
| Zoning | 100 | 1 | 0.015 | 0.015 |
| Zoning | 100 | 5 | 0.096 | 0.042 |
| Zoning | 100 | 10 | 0.159 | 0.050 |
| Zoning | 100 | 20 | 0.215 | 0.054 |
| Zoning | 100 | 100 | 0.489 | 0.059 |

The open-source question-answering does not work particularly well on the policy surveillance datasets, likely because the language used in asking factual questions about information in Wikipedia articles is very different from the language found in legal text. Surprisingly, the system performs better with only 20 retrieved documents to extract an answer; for example, on the Complete Streets data, the accuracy at K=20 was 0.218 with 20 documents but 0.145 with 100 documents. Even though when the system has 100 retrieved documents it is more likely to have the correct document as an option from which to extract text, the system is instead more likely to generate multiple incorrect answers from incorrect documents. Therefore, in our further experimentation, we only retrieved 20 documents.

Results of the trained and fine-tuned question-answering systems on the test sets for the Complete Streets dataset are shown in Table 16. Results of the trained and fine-tuned question-answering systems on the test sets for the Local Inclusionary Zoning dataset are shown in Table 17.

Table 16. Question-Answering Results for the Complete Streets Test Dataset

| **Method** | **Training Dataset** | **K** | **Accuracy** | **MRR** |
| --- | --- | --- | --- | --- |
| Trained QA System | Small Dataset (5 states) | 1 | 0.076 | 0.076 |
| Trained QA System | Small Dataset (5 states) | 5 | 0.260 | 0.148 |
| Trained QA System | Small Dataset (5 states) | 10 | 0.293 | 0.153 |
| Trained QA System | Small Dataset (5 states) | 20 | 0.423 | 0.161 |
| Trained QA System | Large Dataset (10 states) | 1 | 0.048 | 0.048 |
| Trained QA System | Large Dataset (10 states) | 5 | 0.187 | 0.099 |
| Trained QA System | Large Dataset (10 states) | 10 | 0.233 | 0.106 |
| Trained QA System | Large Dataset (10 states) | 20 | 0.354 | 0.114 |
| Finetuned QA System | Small Dataset (5 states) | 1 | 0.060 | 0.060 |
| Finetuned QA System | Small Dataset (5 states) | 5 | 0.212 | 0.110 |
| Finetuned QA System | Small Dataset (5 states) | 10 | 0.284 | 0.119 |
| Finetuned QA System | Small Dataset (5 states) | 20 | 0.332 | 0.123 |
| Finetuned QA System | Large Dataset (10 states) | 1 | 0.079 | 0.079 |
| Finetuned QA System | Large Dataset (10 states) | 5 | 0.242 | 0.134 |
| Finetuned QA System | Large Dataset (10 states) | 10 | 0.317 | 0.144 |
| Finetuned QA System | Large Dataset (10 states) | 20 | 0.369 | 0.148 |

Table 17. Question-Answering Results for the Local Inclusionary Zoning Test Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **K** | **Accuracy** | **MRR** |
| Trained QA System | 1 | 0.030 | 0.030 |
| Trained QA System | 5 | 0.189 | 0.082 |
| Trained QA System | 10 | 0.511 | 0.123 |
| Trained QA System | 20 | 0.878 | 0.151 |
| Finetuned QA System | 1 | 0.048 | 0.048 |
| Finetuned QA System | 5 | 0.307 | 0.139 |
| Finetuned QA System | 10 | 0.385 | 0.150 |
| Finetuned QA System | 20 | 0.667 | 0.169 |

We can see that the question-answering systems did reasonably well on the Local Inclusionary Zoning dataset but did not do so well on the Complete Streets dataset. For example, the trained QA system achieved 0.878 accuracy at K=20 on the Local Inclusionary Zoning data, while the corresponding system only achieved 0.423 accuracy on the Complete Streets data. Interestingly, the models trained from scratch outperformed those that were finetuned starting from the open-source model; for example, the model trained on the Local Inclusionary Zoning data achieved 0.878 accuracy at K=20, while the finetuned model achieved 0.667 accuracy. Perhaps, given the small size of the datasets, there was not enough data to allow the model to “unlearn” the features of the SQuAD dataset and learn the features of legal text.

There are likely several reasons that the models performed better on the Inclusionary Zoning dataset than they did on the Complete Streets dataset. One is that the municipalities in the Local Inclusionary Zoning study have smaller bodies of law than the states in the Complete Streets study, so the BM-25 document retriever feeding the question-answering system performs better, since it is simply numerically easier to find the correct legal documents. A second reason is that many of the questions in the Local Inclusionary Zoning dataset, such as “What is the required set-aside?” and “What is the development size threshold?” are asking for numerical facts as answers. The SQuAD dataset contains many factoid questions in this style, and the trained models performed particularly well on these questions. The Complete Streets dataset does not contain questions with numerical answers like this. Finally, during the training process, each model was trained for 40 epochs, or passes through the training data, and the model from the epoch with the highest accuracy on the validation set was chosen. For the Complete Streets data, the best scoring models occurred after 4-10 epochs, but for the Local Inclusionary Zoning data, it took 28-36 epochs to achieve the highest accuracy. Typically, you do not want to train a model for so long, as it is likely to simply start “memorizing” features of its training data, rather than learning patterns it can generalize to unseen data. However, in this case, since there is a very small amount of training data, it may be beneficial to train the model for a longer time.

Evidence that this is in fact the case comes from examining what happens if we take a model that we trained on one of the policy surveillance datasets and evaluate it on the other dataset. If, in training the model for many epochs, the model has simply memorized its training data, we should see poor performance on the other dataset. However, if the model just took a long time to learn meaningful features due to the small size of its training data, it should perform at a higher level on completely novel data. Table 18 shows the results of evaluating a question-answering system trained on one policy surveillance dataset on the other policy surveillance dataset.

Table 18. Question-Answering Results for Models Evaluated on a Different Dataset

| **Method** | **Training Dataset** | **K** | **Evaluation Dataset** | **Accuracy** | **MRR** |
| --- | --- | --- | --- | --- | --- |
| Trained QA System | Complete Streets Small (5 states) | 1 | Zoning | 0.037 | 0.037 |
| Trained QA System | Complete Streets Small (5 states) | 5 | Zoning | 0.119 | 0.067 |
| Trained QA System | Complete Streets Small (5 states) | 10 | Zoning | 0.174 | 0.073 |
| Trained QA System | Complete Streets Small (5 states) | 20 | Zoning | 0.519 | 0.096 |
| Trained QA System | Complete Streets Large (10 states) | 1 | Zoning | 0.033 | 0.033 |
| Trained QA System | Complete Streets Large (10 states) | 5 | Zoning | 0.082 | 0.047 |
| Trained QA System | Complete Streets Large (10 states) | 10 | Zoning | 0.159 | 0.059 |
| Trained QA System | Complete Streets Large (10 states) | 20 | Zoning | 0.441 | 0.077 |
| Trained QA System | Zoning | 1 | Complete Streets | 0.051 | 0.051 |
| Trained QA System | Zoning | 5 | Complete Streets | 0.151 | 0.086 |
| Trained QA System | Zoning | 10 | Complete Streets | 0.326 | 0.108 |
| Trained QA System | Zoning | 20 | Complete Streets | 0.677 | 0.133 |
| Finetuned QA System | Complete Streets Small (5 states) | 1 | Zoning | 0.059 | 0.059 |
| Finetuned QA System | Complete Streets Small (5 states) | 5 | Zoning | 0.182 | 0.103 |
| Finetuned QA System | Complete Streets Small (5 states) | 10 | Zoning | 0.282 | 0.117 |
| Finetuned QA System | Complete Streets Small (5 states) | 20 | Zoning | 0.363 | 0.122 |
| Finetuned QA System | Complete Streets Large (10 states) | 1 | Zoning | 0.059 | 0.059 |
| Finetuned QA System | Complete Streets Large (10 states) | 5 | Zoning | 0.156 | 0.096 |
| Finetuned QA System | Complete Streets Large (10 states) | 10 | Zoning | 0.244 | 0.108 |
| Finetuned QA System | Complete Streets Large (10 states) | 20 | Zoning | 0.296 | 0.112 |
| Finetuned QA System | Zoning | 1 | Complete Streets | 0.060 | 0.060 |
| Finetuned QA System | Zoning | 5 | Complete Streets | 0.163 | 0.093 |
| Finetuned QA System | Zoning | 10 | Complete Streets | 0.293 | 0.110 |
| Finetuned QA System | Zoning | 20 | Complete Streets | 0.432 | 0.120 |

Here we see that, while the question-answering systems trained on the Local Inclusionary Zoning data do not perform as well on the Complete Streets data as they did on the Local Inclusionary Zoning data, they actually outperform the models listed in Table 16 that are trained directly on the Complete Streets data.[[26]](#footnote-27) For example, the model trained on the Local Inclusionary Zoning data achieves 0.878 accuracy at K=20 on the Local Inclusionary Zoning data, while achieving a lower accuracy of 0.677 at K=20 on the Complete Streets data. However, this is better than the model trained directly on the Complete Streets data, which only achieves an accuracy of 0.423 at K=20. This indicates that these question-answering systems did in fact need a lot of passes through the small dataset to learn relevant features, and the models trained on the Complete Streets data simply were not trained for long enough. It is possible that choosing a model for the Complete Streets data from a later epoch in the training process would perform better on the test data.

Despite this potential, we do not recommend further exploration of question-answering systems for the policy surveillance process. It appears that in order for a question answering system to work well, it needs to be trained for a large number of epochs, but it is unclear how we would know which specific instance of the model to select. The typical method of selecting a model is to choose the model that has the best accuracy on the validation set. In this experimentation, that method indicated that a model from much earlier in the training process would be the best one, but the transfer learning experimentation showed that models from much later in the training process achieved greater generalizability. Furthermore, even in the best case, the question-answering systems only achieved high accuracy after listing 20 possible answers, and policy surveillance practitioners would likely prefer not to look through that many answers before arriving at the correct one.

**Recommendation:** We do not recommend further exploration of question-answering systems for the policy surveillance process. It seems that a quality document retrieval system could obviate the need for a separate question-answering system.

The document retrieval methods we experimented with identified 200-word subsections of the legal texts. This is not much longer than many of the extracts identified as containing the answers to the policy surveillance questions. It is largely unnecessary to further identify one or two sentences when we have already identified such a short relevant subsection of a document. If these 200-word sections are still too long to be useful, we would recommend performing the document search methods over 100 or 150-word subsections of the legal texts, and that would perform much the same function as a question-answering system.

4.4 How Do We Extract Answers to Questions?

Once the relevant span of text to answer a question has been identified, it needs to be converted into a binary or categorical answer to a policy surveillance question. We experimented with two approaches to do so. In section 4.4.1, we explored using existing textual entailment models to arrive at an answer to a question. In section 4.4.2, we trained binary classifiers to answer the policy surveillance questions.

4.4.1 Textual Entailment Models

Textual entailment[[27]](#footnote-28) seeks to determine whether the truth of one statement follows from another. This approach can be used to determine whether one statement entails, contradicts, or is neutral toward another statement, and can support automated question answering. In the case of policy surveillance, we are given a policy surveillance question and the text identified as providing support to the answer as described in section 3.2. We can then rewrite the question in a statement form. For example, Question 5, “Must information about Complete Streets implementation be publicly released?” can be written as “Information about Complete Streets implementation must be publicly released.” We can then apply a textual entailment model to the paired rewritten question and the identified supporting text. If the textual entailment model predicts “entailment”, meaning the supporting text entails that the question rewritten as a statement is true, that would imply that the answer to the policy surveillance question is “yes”. If the textual entailment model predicts “contradiction”, that would imply that the answer to the policy surveillance question is “no”.

We examined how well the Allen NLP[[28]](#footnote-29) textual entailment model performs on both the Complete Streets and Local Inclusionary Zoning datasets. The Allen NLP textual entailment model is trained on the Stanford Natural Language Inference (SNLI) Corpus[[29]](#footnote-30), which contains 570,000 English sentence pairs on general subjects. Table 19 shows the results of evaluating the Allen NLP textual entailment model on the test sets of the policy surveillance studies. For input to the model, we experimented with using both the full text extract as identified by the policy surveillance researchers, and the shorter excerpt from the legal text directly supporting the answer to the question as described in section 3.2. We report accuracy, precision (the fraction of predicted “yes” answers that are correct), recall (the fraction of true “yes” answers that are predicted to be “yes”), and F1-score (the harmonic mean of precision and recall).

Table 19. Allen NLP Textual Entailment Models on Policy Surveillance Datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dataset** | **Input Text** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| Complete Streets | Full Text | 0.575 | 0.529 | 0.878 | 0.660 |
| Complete Streets | Short Text | 0.536 | 0.504 | 0.769 | 0.609 |
| Zoning | Full Text | 0.478 | 0.391 | 0.904 | 0.547 |
| Zoning | Short Text | 0.430 | 0.360 | 0.819 | 0.500 |

We can see that the results are quite poor overall. The high recall and low precision shows that the models usually predicted entailment, regardless of the specific input. The poor performance is likely due to the fact that the Allen NLP models are trained on the SNLI corpus, which includes very simple sentences describing common activities (an example sentence pair is “A soccer game with multiple males playing”, which entails “Some men are playing a sport”.) This language is very different from legal text, and the model is unable to correctly analyze the legal language.

**Recommendation**: We do not recommend using this textual entailment model in future policy surveillance work.

4.4.2 Binary Classifiers

Policy surveillance questions can be classified as binary (yes/no), categorical – mutually exclusive (choose one answer from three or more answer choices), or categorical – select all that apply. (It may in theory be possible for a policy surveillance question to have a free text answer; however, using this question type is not recommended, and it is not used in either the Complete Streets or Local Inclusionary Zoning studies. Therefore, we did not explore methods for automatically answering questions of this type.) As discussed in Section 3.2, we converted the “categorical – select all that apply” questions into multiple binary questions. We then employed multiple text classifiers of varying complexity and sophistication to train machine learning systems to answer a diverse range of binary questions. For input to the models, we concatenated the text of the binary question and the shorter excerpt of the legal text supporting the answer. For the neural network models, only the shorter version of the text was used due to the memory constraints inherent in running these large models. We experimented with the following models:

* A logistic regression classifier, which is simple to implement but just relies on the presence of certain words in the text and does not take into account the sequence or context of the words in the text. For example, a logistic regression model would treat the word “bank” identically, regardless of whether it appeared in the phrase “river bank” or “bank account”. For the logistic regression classifier, we experimented with using phrases ranging from 1-4 words long, omitting the least frequent words (which might be too specific to one example and not useful for generalization), omitting the most frequent words (common words like “the” and “and”, which may not contain useful information), and interpreting the features as binary (does the input contain this word or not) versus numeric (how many times does this word appear). We report the best results, which occurred using unigrams (one-word phrases) with the shorter text and trigrams (three-word phrases) with the full text. For both input formats, the best results were with binary features, not omitting any infrequent words, and omitting terms that appear in more than 75% of examples.
* A recurrent neural network, which uses the sequence of words in the text as input, so it retains contextual information that can assist in disambiguating polysemous terms. For recurrent neural networks, we experimented with both a convolutional neural network (CNN) and an LSTM (Long Short-Term Memory) network. A CNN looks at subsequences of the textual input using a sliding window approach, so the phrase “In Covered Development Projects that contain rental units…” would be analyzed by looking at the subphrases such as “In Covered Development”, “Development Projects that”, and “that contain rental”, where the specific length of the subphrases and the number of overlapping terms between each subphrase are parameters of the model. In contrast, an LSTM looks at an entire longer sequence at once and tries to infer dependencies between the various terms.
* A neural network with an attention layer between the question and the extracted sentence. Attention is a technique used in neural networks that places more emphasis on the words in the text that are the most important. When you use attention to compare a question to an extracted sentence, it identifies which words in the extracted text correlate to words in the question. We experimented with whether the supporting text or the question should be first in the input and found the best results when the supporting text was first.
* A neural network based on BERT to predict a yes or no answer. Since BERT was pretrained on 3.3 billion words of English data, it has encoded a large amount of linguistic and real-world knowledge, and that knowledge can be used to help understand textual input. We trained models based on both BERT and RoBERTa[[30]](#footnote-31), a modified version of BERT that was pretrained with more data for more epochs.

The logistic regression classifiers were implemented using Scikit-Learn,[[31]](#footnote-32) an open-source library for implementing machine learning algorithms in Python. The neural networks using BERT and RoBERTa were implemented with Huggingface Transformers,[[32]](#footnote-33) an API for using pre-trained machine learning models. All other neural networks were implemented using Keras[[33]](#footnote-34), an API for implementing neural networks in Python. Table 20 shows the results of evaluating all of these classifiers on the test sets of the policy surveillance studies. For training and validating all classifiers, we used the data splits described in Section 3.3. We report accuracy, precision, recall, and F1-score.

Table 20. Binary Classifiers on Policy Surveillance Datasets

| **Model Type** | **Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- | --- | --- |
| Logistic Regression (Short Text) | Complete Streets Small (5 states) | 0.806 | 0.740 | 0.910 | 0.805 |
| Logistic Regression (Full Text) | Complete Streets Small (5 states) | 0.830 | 0.784 | 0.885 | 0.830 |
| CNN | Complete Streets Small (5 states) | 0.813 | 0.745 | 0.917 | 0.822 |
| LSTM | Complete Streets Small (5 states) | 0.837 | 0.807 | 0.859 | 0.832 |
| Attention | Complete Streets Small (5 states) | 0.831 | 0.791 | 0.872 | 0.829 |
| BERT | Complete Streets Small (5 states) | 0.861 | 0.842 | 0.909 | 0.874 |
| RoBERTa | Complete Streets Small (5 states) | 0.837 | 0.839 | 0.858 | 0.848 |
| Logistic Regression (Short Text) | Complete Streets Large (10 states) | 0.842 | 0.795 | 0.897 | 0.842 |
| Logistic Regression (Full Text) | Complete Streets Large (10 states) | 0.839 | 0.794 | 0.891 | 0.839 |
| CNN | Complete Streets Large (10 states) | 0.831 | 0.813 | 0.833 | 0.823 |
| LSTM | Complete Streets Large (10 states) | 0.849 | 0.816 | 0.878 | 0.846 |
| Attention | Complete Streets Large (10 states) | 0.846 | 0.800 | 0.897 | 0.846 |
| BERT | Complete Streets Large (10 states) | 0.858 | 0.882 | 0.847 | 0.864 |
| RoBERTa | Complete Streets Large (10 states) | 0.864 | 0.902 | 0.835 | 0.867 |
| Logistic Regression (Short Text) | Zoning | 0.715 | 0.610 | 0.500 | 0.671 |
| Logistic Regression (Full Text) | Zoning | 0.737 | 0.709 | 0.415 | 0.671 |
| CNN | Zoning | 0.741 | 0.740 | 0.394 | 0.514 |
| LSTM | Zoning | 0.759 | 0.809 | 0.404 | 0.539 |
| Attention | Zoning | 0.752 | 0.814 | 0.372 | 0.511 |
| BERT | Zoning | 0.848 | 0.857 | 0.920 | 0.888 |
| RoBERTa | Zoning | 0.852 | 0.886 | 0.886 | 0.886 |

Across all datasets, we can see that the BERT and RoBERTa models performed the best, showing that the classifier is able to leverage the large amount of linguistic knowledge from pretraining and apply it to this domain. The roughly equivalent precision and recall scores show that the model is truly learning to predict an answer and not defaulting to either a “yes” or a “no” answer. The LSTM model was the next best performing model, as its handling of long sequences is ideal for textual input. Somewhat surprisingly, the LSTM model outperformed the attention-based models. Perhaps because the policy surveillance questions are all relatively short, the attention mechanism drawing focus to the most important part of the question is unnecessary. The CNN model did not do much better than the logistic regression models that only used word-level features and did not take the sequence of the text into account. This is not surprising, as the CNN’s sliding window approach is typically more suited to two-dimensional image data (for example, labelling objects in images).

In the Complete Streets data in Table 20, we found that all of the models trained on 10 states’ data outperform the models trained on 5 states’ data, illustrating the importance of a large amount of training data for neural models. However, the best models trained on only 5 states’ data still achieve over 85% in both accuracy and F1-score. These results are likely accurate enough to serve as a good first analysis, which a policy surveillance practitioner can subsequently review and correct when necessary.

In the Local Inclusionary Zoning data, we also see significantly better performance from the BERT and RoBERTa models than we do from the other models. The other models have high precision and low recall, indicating that they usually predict a “no” answer to the policy surveillance question. In other words, when the models predict “yes” they are correct in doing so, but they fail to predict a “yes” answer in many cases when they should. We did not see this happen in the Complete Streets data. A likely explanation is that the Local Inclusionary Zoning dataset includes many questions about numeric attributes, such as thresholds for development or income requirements. This meant that after converting the questions into binary format there were many pairs of questions and supporting text that were identical except for the specific number used (i.e., “Is the income target for rental units up to 65% median income?”, “Is the income target for rental units up to 80% median income?”, etc.) Numbers are very difficult for a machine learning system to distinguish, and it is understandable that the system would choose to maximize accuracy by predicting “no” as the answer to all question-text pairs, since “no” is the correct answer for all but one of the questions in the set. The more sophisticated BERT and RoBERTa models likely had a better underlying representation of numeric values due to their pretraining process and were better able to handle these questions.

While these models perform quite well on the test sets corresponding to their training data, it would be most useful if they had learned more general features of legal language and could be applied to novel policy surveillance datasets. Table 21 shows the results of evaluating the BERT and RoBERTa models trained on one policy surveillance dataset on the other policy surveillance dataset.

Table 21. Binary Classification Results for Models Evaluated on a Different Dataset

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Training Dataset** | **Evaluation Dataset** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| BERT | Complete Streets Small (5 states) | Zoning | 0.419 | 0.368 | 0.936 | 0.529 |
| RoBERTa | Complete Streets Small (5 states) | Zoning | 0.678 | 0.717 | 0.835 | 0.772 |
| BERT | Complete Streets Large (10 states) | Zoning | 0.489 | 0.397 | 0.904 | 0.552 |
| RoBERTa | Complete Streets Large (10 states) | Zoning | 0.419 | 0.359 | 0.851 | 0.505 |
| BERT | Zoning | Complete Streets | 0.633 | 0.663 | 0.625 | 0.643 |
| RoBERTa | Zoning | Complete Streets | 0.684 | 0.609 | 0.917 | 0.731 |

The models do not demonstrate great transfer performance to a different dataset. The models trained on the Complete Streets dataset typically demonstrated low precision and high recall on the Local Inclusionary Zoning dataset, indicating that they typically predicted a “yes” answer for those questions, regardless of whether the answer should be “yes” or “no”. Since the Complete Streets data did not include any questions about numeric attributes, it is likely that the model had trouble distinguishing between the multiple versions of a question that differed only in the numerical value. In these instances, since the supporting text still generally corresponded to the meaning of the question, the model predicted a “yes” answer to the question even if the numbers did not match. (The RoBERTa model trained on the smaller Complete Streets dataset does significantly better on the Local Inclusionary Zoning dataset than the other models. It is unclear why this is the case, except to note that this model had the lowest score on the training dataset of all the models presented, so perhaps there was something slightly awry in the original training of the model.) The models trained on the Local Inclusionary Zoning data did a somewhat better job generalizing to the Complete Streets data since they were not confronted with novel questions about numerical thresholds, but still fell short of the performance of the models trained directly on the Complete Streets data, with the RoBERTa model achieving accuracy of 0.684 compared to the RoBERTa model trained on the Complete Streets data, which achieved accuracy of 0.864. This is likely short of the performance needed to be useful to a policy surveillance practitioner.

Since the BERT and RoBERTa models were so successful when evaluated on the test dataset corresponding to their training data, we recommend pursuing their use in converting legal text into a binary answer to a policy surveillance question.

**Recommendation**: To work toward a model that will be applicable across a wide range of policy surveillance topics, we recommend jointly training BERT and RoBERTa models on multiple policy surveillance datasets at once. This way the model will see a larger variety of question types during its training and be better able to generalize to new topics. Alternatively, a policy surveillance researcher could manually answer questions for the first five jurisdictions in a given study, and then a classifier could be trained to assist in answering the questions for the remaining jurisdictions. This would still greatly speed up the policy surveillance process, even if the work on the first few jurisdictions was still done without automation.

To use a BERT or RoBERTa classifier, all of the questions must be written in binary format.

**Recommendation**: We suggest that going forward, policy surveillance researchers formulate all of their questions in a binary format during the research process, even if they start out writing a categorical question when they initially formulate the question. If it is too unwieldy for an analyst reviewing the policy surveillance study to read a long list of binary questions, the questions can be converted into a categorical format prior to publication.

5 Recommendations

**Potential for artificial intelligence and natural language processing in policy surveillance**

The results of this experimentation point toward a path of using artificial intelligence and natural language processing to aid in the process of policy surveillance. These tools would not eliminate the role of the policy surveillance researcher but would provide assistance to make the process less labor-intensive.

Figure 4 illustrates the steps of the policy surveillance process, indicating which steps would be augmented by particular technologies.

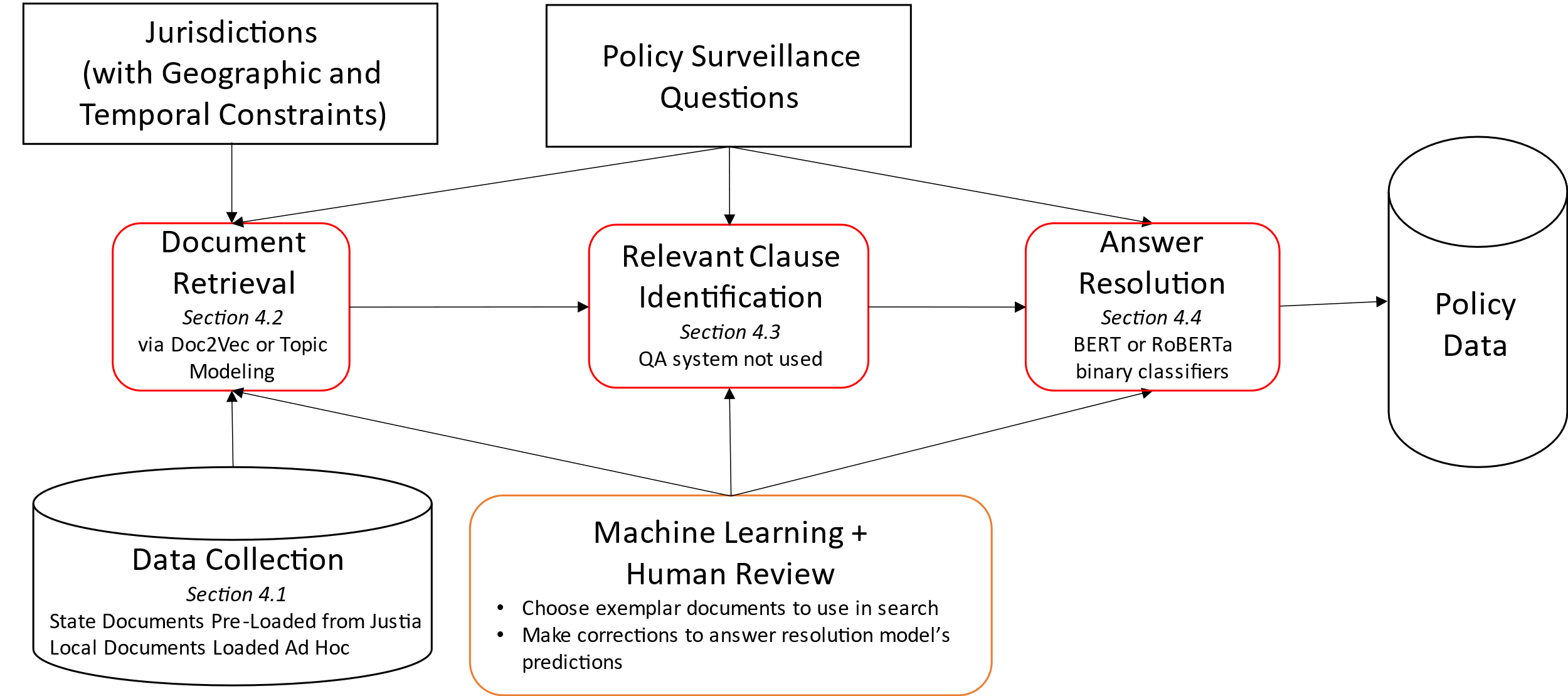


Figure 4: Policy Surveillance Process with Recommendations of How to Incorporate Artificial Intelligence

**Maintain a database of state-level statutes; Obtain local statutes on an ad hoc basis**

In preparation for a policy surveillance study, the relevant legal documents would need to be gathered. A computer program could automatically crawl Justia and parse the documents to populate a database of state codes and regulations. The code for municipal jurisdictions, as well as any state-level legal documents that are not part of the state code (such as policy documents from state agencies) would need to be individually acquired and processed. These documents would then be automatically segmented into 100 to 200-word excerpts, and a document vector model (or potentially a topic model) would be trained over these excerpts.

**Utilize document embeddings to identify relevant documents; continue experimenting with using a topic-model based search**

Once complete, a policy surveillance researcher could start searching for the relevant legal documents using standard keyword search techniques. The researcher could augment their initial search by using word embeddings trained on the legal documents to identify more search terms that might point to additional relevant documents. Once the researcher has identified the relevant documents for an initial jurisdiction, they can use the document vector model (or potentially the topic model) to identify documents with similar content. This could be used to find additional documents associated with the original jurisdiction of interest, or to find similar legal documents in different jurisdictions. While the documents identified will be 100 to 200-word excerpts of the complete legal text, one can envision a user interface that would allow the researcher to read the excerpts in their larger context.

**Convert all policy surveillance questions to binary format and use BERT and RoBERTa-based binary classifiers to perform answer classification.**

The system could then combine this legal text with the text of the specific policy surveillance question of interest (provided by the policy surveillance researcher) and apply a BERT or RoBERTa-based classifier to predict a yes or no answer to that question. The BERT or RoBERTa-based classifier would have been trained in advance on a large number of previously completed policy surveillance studies. The policy surveillance researcher would review the answer predicted by the system for correctness. If the system was incorrect, the policy surveillance researcher could indicate as such and the classifier could be retrained, improving the predictions for future policy surveillance studies.

**NOTICE**

This report was produced for the U. S. Government under Contract Number 75D30120F09743, and is subject to Federal Acquisition Regulation Clause 52.227-14, Rights in Data-General.

No other use other than that granted to the U. S. Government, or to those acting on behalf of the U. S. Government under that Clause is authorized without the express written permission of The MITRE Corporation.

For further information, please contact The MITRE Corporation, Contracts Management Office, 7515 Colshire Drive, McLean, VA 22102-7539, (703) 983-6000.

**© 2023 The MITRE Corporation.**

1. Burris, S, JD; Cloud, LK, JD; Penn, M, JD, MLIS. The Growing Field of Legal Epidemiology, Journal of Public Health Management and Practice: March/April 2020 – Volume 26 – Issue – pp S4-S9.  [↑](#footnote-ref-2)
2. "Complete Streets." In: The Policy Surveillance Program, 2020. https:// lawatlas.org/datasets/complete-streets [↑](#footnote-ref-3)
3. “Local Inclusionary Zoning Laws.” In: The Policy Surveillance Program, 2019. https://lawatlas.org/datasets/inclusionary-zoning [↑](#footnote-ref-4)
4. Pincites are references to specify or pinpoint the specific section of relevant legal text. [↑](#footnote-ref-5)
5. JavaScript Object Notation (JSON) is a lightweight data interchange format based on lists and key-value pairs that is both easy for humans to read and easy for machines to parse. [↑](#footnote-ref-6)
6. A hash value is a numeric value that is used to uniquely identify a piece of data. [↑](#footnote-ref-7)
7. Other policy surveillance studies may review legislation and case law, but these were not used in the Complete Streets or Local Inclusionary Zoning studies. [↑](#footnote-ref-8)
8. https://public.resource.org/ [↑](#footnote-ref-9)
9. AR, CO, GA, ID, KY, MS, TN, VT, VA, WY. [↑](#footnote-ref-10)
10. Individuals appear to have designed APIs for California’s and DC’s statutes, but these resources are not current. [↑](#footnote-ref-11)
11. AR, GA, MS, and TN do not host the text of their state statutes themselves, but instead direct readers to Lexis. [↑](#footnote-ref-12)
12. https://tika.apache.org/ [↑](#footnote-ref-13)
13. https://github.com/deepset-ai/haystack [↑](#footnote-ref-14)
14. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. [↑](#footnote-ref-15)
15. Sugathadasa, K., Ayesha, B., de Silva, N., Perera, A. S., Jayawardana, V., Lakmal, D., & Perera, M. (2017, December). Synergistic union of doc and lexicon for domain specific semantic similarity. In *2017 IEEE international conference on industrial and information systems (ICIIS)* (pp. 1-6). IEEE. [↑](#footnote-ref-16)
16. https://archive.org/details/Law2Vec [↑](#footnote-ref-17)
17. Le, Q., & Mikolov, T. (2014, June). Distributed representations of sentences and documents. In *International conference on machine learning* (pp. 1188-1196). PMLR. [↑](#footnote-ref-18)
18. https://tmnt.readthedocs.io/en/latest/ [↑](#footnote-ref-19)
19. Kingma DP, Welling M. Auto-Encoding Variational Bayes. In: Proceedings of the 2nd International Conference on Learning Representations (ICLR). IEEE (2014). [↑](#footnote-ref-20)
20. Rajaraman, A.; Ullman, J.D. (2011). "Data Mining" (PDF). Mining of Massive Datasets. pp. 1–17. doi:10.1017/CBO9781139058452.002. ISBN 978-1-139-05845-2. [↑](#footnote-ref-21)
21. Robertson, S., & Zaragoza, H. (2009). *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc. [↑](#footnote-ref-22)
22. Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., ... & Yih, W. T. (2020). Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*. [↑](#footnote-ref-23)
23. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. [↑](#footnote-ref-24)
24. Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. Transactions of the Association of Computational Linguistics (TACL). [↑](#footnote-ref-25)
25. Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*. [↑](#footnote-ref-26)
26. A similar evaluation was performed for the dense passage retrievers, where the retrievers trained on the Complete Streets dataset were evaluated on the Local Inclusionary Zoning dataset, and vice versa. Since the retriever accuracy was low, the accuracy on the other dataset was very low, and therefore not discussed further. [↑](#footnote-ref-27)
27. https://aclanthology.org/D16-1244.pdf [↑](#footnote-ref-28)
28. https://allenai.org/allennlp [↑](#footnote-ref-29)
29. [Samuel R. Bowman](https://cims.nyu.edu/~bowman/), [Gabor Angeli](http://cs.stanford.edu/~angeli/), [Christopher Potts](http://www.stanford.edu/~cgpotts/), and [Christopher D. Manning](http://nlp.stanford.edu/~manning/). 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. [↑](#footnote-ref-30)
30. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*. [↑](#footnote-ref-31)
31. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830. [↑](#footnote-ref-32)
32. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020, October). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations* (pp. 38-45). [↑](#footnote-ref-33)
33. Chollet, F., & others. (2015). Keras. Retrieved from https://keras.io. [↑](#footnote-ref-34)