A Dataset for Statutory Reasoning in Tax Law Entailment and Question Answering

Nils Holzenberger
Johns Hopkins University
Baltimore, Maryland, USA
nilsh@jhu.edu

Andrew Blair-Stanek
U. of Maryland Carey School of Law
Baltimore, Maryland, USA
Johns Hopkins University
Baltimore, Maryland, USA
ablair-stanek@law.umaryland.edu

Benjamin Van Durme
Johns Hopkins University
Baltimore, Maryland, USA
vandurme@cs.jhu.edu

ABSTRACT

Legislation can be viewed as a body of prescriptive rules expressed in natural language. The application of legislation to facts of a case we refer to as statutory reasoning, where those facts are also expressed in natural language. Computational statutory reasoning is distinct from most existing work in machine reading, in that much of the information needed for deciding a case is declared exactly once (a law), while the information needed in much of machine reading tends to be learned through distributional language statistics. To investigate the performance of natural language understanding approaches on statutory reasoning, we introduce a dataset, together with a legal-domain text corpus. Straightforward application of machine reading models exhibits low out-of-the-box performance on our questions, whether or not they have been fine-tuned to the legal domain. We contrast this with a hand-constructed Prolog-based system, designed to fully solve the task. These experiments support a discussion of the challenges facing statutory reasoning moving forward, which we argue is an interesting real-world task that can motivate the development of models able to utilize prescriptive rules specified in natural language.

CCS CONCEPTS

• **Applied computing** → **Law**; • **Computing methodologies** → *Natural language processing*; *Knowledge representation and reasoning*.

KEYWORDS

Law, NLP, Reasoning, Prolog

ACM Reference Format:
Nils Holzenberger, Andrew Blair-Stanek, and Benjamin Van Durme. 2020. A Dataset for Statutory Reasoning in Tax Law Entailment and Question Answering. In *Proceedings of the 2020 Natural Legal Language Processing (NLLP) Workshop, 24 August 2020, San Diego, US.* ACM, New York, NY, USA, 8 pages.

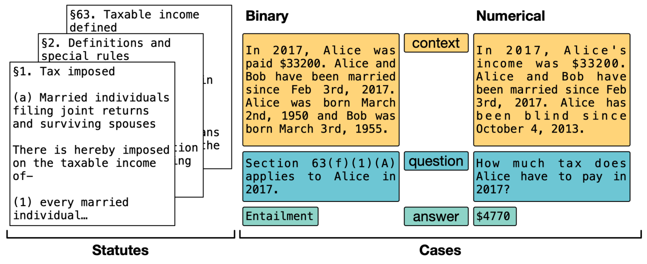
1 INTRODUCTION

Early artificial intelligence research focused on highly-performant, narrow-domain reasoning models, for instance in health [37, 40, 54] and law [30, 38]. Such *expert systems* relied on hand-crafted inference rules and domain knowledge, expressed and stored with the formalisms provided by databases [21]. The main bottleneck of this approach is that experts are slow in building such knowledge bases and exhibit imperfect recall, which motivated research into models for automatic information extraction (e.g. Lafferty et al. [36]). Systems for large-scale automatic knowledge base construction have improved (e.g. Etzioni et al. [20], Mitchell et al. [41]), as well as systems for sentence level semantic parsing [64]. Among others, this effort has led to question-answering systems for games [22] and, more recently, for science exams [14, 23, 27]. The challenges include extracting ungrounded knowledge from semi-structured sources, e.g. textbooks, and connecting high-performance symbolic solvers with large-scale language models.

In parallel, models have begun to consider task definitions like Machine Reading (MR) [46] and Recognizing Textual Entailment (RTE) [15, 16] as not requiring the use of explicit structure. Instead, the problem is cast as one of mapping inputs to high-dimensional, dense representations that implicitly encode meaning [18, 45], and are employed in building classifiers or text decoders, bypassing classic approaches to symbolic inference.

This work is concerned with the problem of statutory reasoning [62, 66]: how to reason about an example situation, a *case*, based on complex rules provided in natural language. In addition to the reasoning aspect, we are motivated by the lack of contemporary systems to suggest legal opinions: while there exist tools to aid lawyers in retrieving relevant documents for a given case, we are unaware of any strong capabilities in automatic statutory reasoning.

Our contributions, summarized in Figure 2, include a novel dataset based on US tax law, together with test cases (Section 2). Decades-old work in expert systems could solve problems of the sort we construct here, based on manually derived rules: we replicate that approach in a Prolog-based system that achieves 100% accuracy on our examples (Section 3). Our results demonstrate that straightforward application of contemporary Machine Reading models is not sufficient for our challenge examples (Section 5), whether or not they were adapted to the legal domain (Section 4). This is meant to provoke the question of whether we should be concerned with: (a) improving methods in semantic parsing in order to replace manual transduction into symbolic form; or (b) improving machine reading methods in order to avoid explicit symbolic solvers. We view this work as part of the conversation including recent work in multi-hop inference [61], where our task is more domain-specific but potentially more challenging.



§63. Taxable income defined

§2. Definitions and special rules

§1. Tax imposed

(a) Married individuals filing joint returns and surviving spouses

There is hereby imposed on the taxable income of–

(1) every married individual…

In 2017, Alice was paid $33200. Alice and Bob have been married since Feb 3rd, 2017. Alice was born March 2nd, 1950 and Bob was born March 3rd, 1955.

Section 63(f)(1)(A) applies to Alice in 2017.

Entailment

In 2017, Alice's income was $33200. Alice and Bob have been married since Feb 3rd, 2017. Alice has been blind since October 4, 2013.

How much tax does Alice have to pay in 2017?

$4770

Figure 1: Sample cases from our dataset. The questions can be answered by applying the rules contained in the statutes to the context.

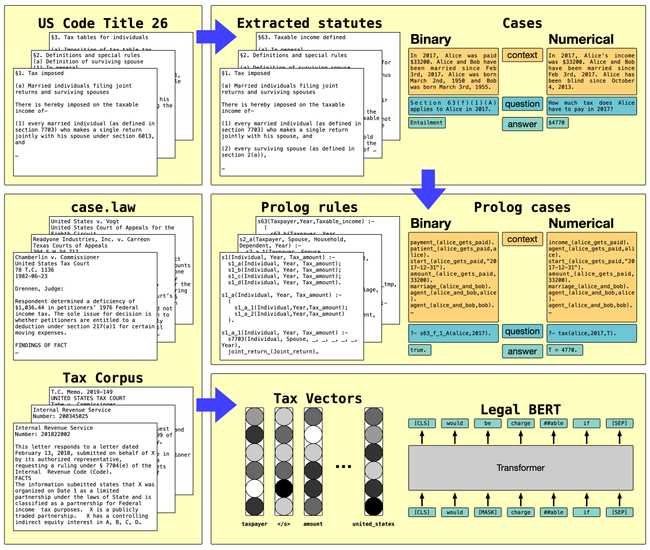


Figure 2: Resources. Corpora on the left hand side were used to build the datasets and models on the right hand side.

2 DATASET

Here, we describe our main contribution, the StAtutory Reasoning Assessment dataset (SARA): a set of rules extracted from the statutes of the US Internal Revenue Code (IRC), together with a set of natural language questions which may only be answered correctly by referring to the rules[1].

The IRC[2] contains rules and definitions for the imposition and calculation of taxes. It is subdvided into sections, which in general, define one or more terms: section 3306 defines the terms employment, employer and wages, for purposes of the federal unemployment tax. Sections are typically structured around a general rule, followed by a number of exceptions. Each section and its subsections may be cast as a predicate whose truth value can be checked against a state of the world. For instance, subsection 7703(a)(2):

> *an individual legally separated from his spouse under a decree of divorce or of separate maintenance shall not be considered as married*

can be checked given an individual.

[1]The dataset can be found under https://nlp.jhu.edu/law/
[2]https://uscode.house.gov/browse/prelim@title26&edition=prelim

Slots are another major feature of the law. Each subsection refers to a certain number of slots, which may be filled by existing entities (in the above, *individual*, *spouse*, and *decree of divorce or of separate maintenance*). Certain slots are implicitly filled: §7703(a)(1) and (b)(3) mention a “spouse", which must exist since the “individual" is married. Similarly, slots which have been filled earlier in the section may be referred to later on. For instance, “household" is mentioned for the first time in §7703(b)(1), then again in §7703(b)(2) and in §7703(b)(3). Correctly resolving slots is a key point in successfully applying the law.

Overall, the IRC can be framed as a set of predicates formulated in human language. The language used to express the law has an open texture [29], which makes it particularly challenging for a computer-based system to determine whether a subsection applies, and to identify and fill the slots mentioned. This makes the IRC an excellent corpus to build systems that reason with rules specified in natural language, and have good language understanding capabilities.

2.1 Statutes and test cases

As the basis of our set of rules, we selected sections of the IRC well-supported by Treasury Regulations, covering tax on individuals (§1), marriage and other legal statuses (§2, 7703), dependents (§152), tax exemptions and deductions (§63, 68, 151) and employment (§3301, 3306). We simplified the sections to (1) remove highly specific sections (e.g. those concerning the employment of sailors) in order to keep the statutes to a manageable size, and (2) ensure that the sections only refer to sections from the selected subset. For ease of comparison with the original statutes, we kept the original numbering and lettering, with no adjustment for removed sections. For example, there is a section 63(d) and a section 63(f), but no section 63(e). We assumed that any taxable year starts and ends at the same time as the corresponding calendar year.

For each subsection extracted from the statutes, we manually created two paragraphs in natural language describing a case, one where the statute applies, and one where it does not. These snippets, formulated as a logical entailment task, are meant to test a system's understanding of the statutes, as illustrated in Figure 1. The cases were vetted by a law professor for coherence and plausibility. For the purposes of machine learning, the cases were split into 176 train and 100 test samples, such that (1) each pair of positive and negative cases belongs to the same split, and (2) each section is split between train and test in the same proportions as the overall split.

Since tax legislation makes it possible to predict how much tax a person owes, we created an additional set of 100 cases where the task is to predict how much tax someone owes. Those cases were created by randomly mixing and matching pairs of cases from the first set of cases, and resolving inconsistencies manually. Those cases are no longer a binary prediction task, but a task of predicting an integer. The prediction results from taking into account the entirety of the statutes, and involves basic arithmetic. The 100 cases were randomly split into 80 training and 20 test samples.

Because the statutes were simplified, the answers to the cases are not those that would be obtained with the current version of the IRC. Some of the IRC counterparts of the statutes in our dataset have been repealed, amended, or adjusted to reflect inflation.

Cross-references	Explicit	Implicit
Within the section	30	25
To another section	34	44

Table 1: Number of subsections containing cross-references

2.2 Key features of the corpus

While the corpus is based on a simplification of the Internal Revenue Code, care was taken to retain prominent features of US law. We note that the present task is only one aspect of legal reasoning, which in general involves many more modes of reasoning, in particular interpreting regulations and prior judicial decisions. The following features are quantified in Tables 1 to 4.

Reasoning with time. The timing of events (marriage, retirement, income...) is highly relevant to determining whether certain sections apply, as tax is paid yearly. In total, 62 sections refer to time. Some sections require counting days, as in §7703(b)(1):

> *a household which constitutes for more than one-half of the taxable year the principal place of abode of a child*

or taking into account the absolute point in time as in §63(c)(7):

> *In the case of a taxable year beginning after December 31, 2017, and before January 1, 2026-*

Exceptions and substitutions. Typically, each section of the IRC starts by defining a general case and then enumerates a number of exceptions to the rule. Additionally, some rules involve applying a rule after substituting terms. A total of 50 sections formulate an exception or a substitution. As an example, §63(f)(3):

> *In the case of an individual who is not married and is not a surviving spouse, paragraphs (1) and (2) shall be applied by substituting "\$750" for "\$600".*

Numerical reasoning. Computing tax owed requires knowledge of the basic arithmetic operations of adding, subtracting, multiplying, dividing, rounding and comparing numbers. 55 sections involve numerical reasoning. The operation to be used needs to be parsed out of natural text, as in §1(c)(2):

> *\$3,315, plus 28% of the excess over \$22,100 if the taxable income is over \$22,100 but not over \$53,500*

Cross-references. Each section of the IRC will typically reference other sections. Table 1 shows how this feature was preserved in our dataset. There are explicit references within the same section, as in §7703(b)(1):

> *an individual who is married (within the meaning of subsection (a)) and who files a separate return*

explicit references to another section, as in §3301:

> *There is hereby imposed on every employer (as defined in section 3306(a)) for each calendar year an excise tax*

and implicit references, as in §151(a), where "taxable income" is defined in §63:

> *the exemptions provided by this section shall be allowed as deductions in computing taxable income.*

Common sense knowledge. Four concepts, other than time, are left undefined in our statutes: (1) kinship, (2) the fact that a marriage ends if either spouse dies, (3) if an event has not ended, then it is ongoing; if an event has no start, it has been true at any time before it ends; and some events are instantaneous (e.g. payments), (4) a person's gross income is the sum of all income and payments received by that person.

	min	max	avg ± stddev	median
Depth of leaf	1	6	3.6 ± 0.8	4
Depth of node	0	6	3.2 ± 1.0	3

Table 2: Statistics about the tree structure of the statutes

Vocabulary size			train	867	statutes	768
			test	535	combined	1596
		min	max	avg	stddev	median
Sentence length (in words)	train	4	138	12.3	9.1	11
	test	4	34	11.6	4.5	10
	statutes	1	88	16.5	14.9	12.5
	combined	1	138	12.7	9.5	11
Case length (in sentences)	train	1	9	4.2	1.7	4
	test	2	7	3.8	1.3	4
	combined	1	9	4.1	1.6	4
Case length (in words)	train	17	179	48.5	22.2	43
	test	17	81	41.6	14.7	38
	combined	17	179	46.3	20.3	41
Section length	sentences	2	16	8.3	4.7	9
	words	62	1151	488.9	310.4	549

Table 3: Language statistics. The word "combined" means merging the corpora mentioned above it.

	min	max	average	stddev	median
train	0	2,242,833	85,804.86	258,179.30	15,506.50
test	0	243,097	65,246.50	78,123.13	26,874.00
combined	0	2,242,833	81,693.19	233,695.33	17,400.50

Table 4: Answers to numerical questions (in \$).

Hierarchical structure. Law statutes are divided into sections, themselves divided into subsections, with highly variable depth and structure. This can be represented by a tree, with a special ROOT node of depth 0 connecting all the sections. This tree contains 132 leaves and 193 nodes (node includes leaves). Statistics about depth are in Table 2.

3 PROLOG SOLVER

It has been shown that subsets of statutes can be expressed in first-order logic, as described in Section 6. As a reaffirmation of this, and as a topline for our task, we have manually translated the statutes into Prolog rules and the cases into Prolog facts, such that each case can be answered correctly by a single Prolog query[3]. The Prolog rules were developed based on the statutes, meaning that the Prolog code clearly reflects the semantics of the textual form, as in Gunning et al. [27]. This is primarily meant as a proof that a carefully crafted reasoning engine, with perfect natural language understanding, can solve this dataset. There certainly are other ways of representing this given set of statutes and cases. The point of this dataset is not to design a better Prolog system, but to help the development of language understanding models capable of reasoning.

3.1 Statutes

Each subsection of the statutes was translated with a single rule, true if the section applies, false otherwise. In addition, subsections define slots that may be filled and reused in other subsections, as described in Section 2. To solve this coreference problem, any term appearing in a subsection and relevant across subsections is turned

[3]The Prolog program can be found under https://nlp.jhu.edu/law/

into an argument of the Prolog rule. The corresponding variable may then be bound during the execution of a rule, and reused in a rule executed later. Unfilled slots correspond to unbound variables.

To check whether a given subsection applies, the Prolog system needs to rely on certain predicates, which directly reflect the facts contained in the natural language descriptions of the cases. For instance, how do we translate *Alice and Bob got married on January 24th, 1993* into code usable by Prolog? We rely on a set of 61 predicates, following neo-davidsonian semantics [9, 17, 42]. The level of detail of these predicates is based on the granularity of the statutes themselves. Anything the statutes do not define, and which is typically expressed with a single word, is potentially such a predicate: marriage, residing somewhere, someone paying someone else, etc. The example above is translated in Figure 3.

3.2 Cases

The natural language description of each case was manually translated into the facts mentioned above. The question or logical entailment prompt was translated into a Prolog query. For instance, *Section 7703(b)(3) applies to Alice maintaining her home for the year 2018.* translates to `s7703_b_3(alice,home,2018).` and *How much tax does Alice have to pay in 2017?* translates to `tax(alice,2017,Amount).`

```
marriage_(alice_and_bob).
agent_(alice_and_bob, alice).
agent_(alice_and_bob, bob).
start_(alice_and_bob, "1993-01-24").
```

Figure 3: Example predicates used.

In the broader context of computational statutory reasoning, the Prolog solver has three limitations. First, producing it requires domain experts, while automatic generation is an open question. Second, translating natural language into facts requires semantic parsing capabilities. Third, small mistakes can lead to catastrophic failure. An orthogonal approach is to replace logical operators and explicit structure with high-dimensional, dense representations and real-valued functions, both learned using distributional statistics. Such a machine learning-based approach can be adapted to new legislation and new domains automatically.

4 LEGAL NLP

As is commonly done in MR, we pretrained our models using two unsupervised learning paradigms on a large corpus of legal text.

4.1 Text corpus

We curated a corpus consisting solely of freely-available tax law documents with 147M tokens. The first half is drawn from cas [1], a project of Harvard's Law Library that scanned and OCR'ed many of the library's case-law reporters, making the text available upon request to researchers. The main challenge in using this resource is that it contains 1.7M U.S. federal cases, only a small percentage of which are on tax law (as opposed to criminal law, breach of contract, bankruptcy, etc.). Classifying cases by area is a non-trivial problem [55], and tax-law cases are litigated in many different courts. We used the heuristic of classifying a case as being tax-law if it met one of the following criteria: the Commissioner of Internal Revenue was a party; the case was decided by the U.S. Tax Court; or, the case was decided by any other federal court, other than a trade tribunal, with the United States as a party, and with the word *tax* appearing in the first 400 words of the case's written opinion.

The second half of this corpus consists of IRS private letter rulings and unpublished U.S. Tax Court cases. IRS private letter rulings are similar to cases, in that they apply tax law to one taxpayer's facts; they differ from cases in that they are written by IRS attorneys (not judges), have less precedential authority than cases, and redact names to protect taxpayer privacy. Unpublished U.S. Tax Court cases are viewed by the judges writing them as less important than those worthy of publication. These were downloaded as PDFs from the IRS and Tax Court websites, OCR'ed with `tesseract` if needed, and otherwise cleaned.

4.2 Tax vectors

Before training a `word2vec` model [39] on this corpus, we did two tax-specific preprocessing steps to ensure that semantic units remained together. First, we put underscores between multi-token collocations that are tax terms of art, defined in either the tax code, Treasury regulations, or a leading tax-law dictionary. Thus, "surviving spouse" became the single token "surviving_spouse". Second, we turned all tax code sections and Treasury regulations into a single token, stripped of references to subsections, subparagraphs, and subclauses. Thus, "Treas. Reg. §1.162-21(b)(1)(iv)" became the single token "sec_1_162_21". The vectors were trained at 500 dimensions using skip-gram with negative sampling. A window size of 15 was found to maximize performance on twelve human-constructed analogy tasks.

4.3 Legal BERT

We performed further training of BERT [18], on a portion of the full `case.law` corpus, including both state and federal cases. We did not limit the training to tax cases. Rather, the only cases excluded were those under 400 characters (which tend to be summary orders with little semantic content) and those before 1970 (when judicial writing styles had become recognizably modern). We randomly selected a subset of the remaining cases, and broke all selected cases into chunks of exactly 510 tokens, which is the most BERT's architecture can handle. Any remaining tokens in a selected case were discarded. Using solely the masked language model task (i.e. not next sentence prediction), starting from `Bert-Base-Cased`, we trained on 900M tokens.

The resulting Legal BERT has the exact same architecture as `Bert-Base-Cased` but parameters better attuned to legal tasks. We applied both models to the natural language questions and answers in the corpus we introduce in this paper. While `Bert-Base-Cased` had a perplexity of 14.4, Legal BERT had a perplexity of just 2.7, suggesting that the further training on 900M tokens made the model much better adapted to legal queries.

We also probed how this further training impacted ability to handle fine-tuning on downstream tasks. The downstream task we chose was identifying legal terms in case texts. For this task, we defined legal terms as any tokens or multi-token collocations that are defined in *Black's Law Dictionary* [25], the premier legal dictionary. We split the legal terms into training/dev/test splits. We put a 4-layer fully-connected MLP on top of both `Bert-Base-Cased` and Legal BERT, where the training objective was `B-I-O` tagging of tokens in 510-token sequences. We trained both on a set of

200M tokens randomly selected from `case.law` cases not previously seen by the model and not containing any of the legal terms in dev or test, with the training legal terms tagged using string comparisons. We then tested both fine-tuned models' ability to identify legal terms from the test split in case law. The model based on `Bert-Base-Cased` achieved F1 = 0.35, whereas Legal BERT achieved F1 = 0.44. As a baseline, two trained lawyers given the same task on three 510-token sequences each achieved F1 = 0.26. These results indicate that Legal BERT is much better adapted to the legal domain than `Bert-Base-Cased`. Black's Law Dictionary has well-developed standards for what terms are or are not included. BERT models learn those standards via the train set, whereas lawyers are not necessarily familiar with them. In addition, pre-processing dropped some legal terms that were subsets of too many others, which the lawyers tended to identify. This explains how BERT-based models could outperform trained humans.

5 EXPERIMENTS

5.1 BERT-based models

In the following, we frame our task as textual entailment and numerical regression. A given entailment prompt q mentions the relevant subsection (as in Figure 1)[4]. We extract s, the text of the relevant subsection, from the statutes. In q, we replace *Section XYZ applies* with *This applies*. We feed the string "[CLS] + s + [SEP] + q + c + [SEP]", where "+" is string concatenation, to BERT [18]. Let r be the vector representation of the token [CLS] in the final layer. The answer (entailment or contradiction) is predicted as $g(\theta_1 \cdot r)$ where θ_1 is a learnable parameter and g is the sigmoid function. For numerical questions, all statutes have to be taken into account, which would exceed BERT's length limit. We encode "[CLS] all [SEP] + q + c + [SEP]" into r and predict the answer as $\mu + \sigma\theta_2 \cdot r$ where θ_2 is a learned parameter, and μ and σ are the mean and standard deviation of the numerical answers on the training set.

For entailment, we use a cross-entropy loss, and evaluate the models using accuracy. We frame the numerical questions as a taxpayer having to compute tax owed. By analogy with the concept of "substantial understatement of income tax" from §6662(d), we define $\Delta(y, \hat{y}) = \frac{|y-\hat{y}|}{\max(0.1y, 5000)}$ where y is the true amount of tax owed, and $\hat{y}$ is the taxpayer's prediction. The case $\Delta(y, \hat{y}) \geq 1$ corresponds to a substantial over- or understatement of tax. We compute the fraction of predictions $\hat{y}$ such that $\Delta(y, \hat{y}) < 1$ and report that as numerical accuracy.[5] The loss function used is:

$$\mathcal{L} = \sum_{i \in I_1} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) + \sum_{i \in I_2} \max(\Delta(y_i, \hat{y}_i) - 1, 0)$$

where I_1 (resp. I_2) is the set of entailment (resp. numerical) questions, y_i is the ground truth output, and $\hat{y}_i$ is the model's output.

We use Adam [34] with a linear warmup schedule for the learning rate. We freeze BERT's parameters, and experiment with unfreezing BERT's top layer. We select the final model based on early stopping with a random 10% of the training examples reserved as a dev set. The best performing model for entailment and for numerical questions are selected separately, during a hyperparameter search around the recommended setting (batch size=32, learning rate=1e-5). To check for bias in our dataset, we drop either the statute, or the context and the statute, in which case we predict the answer from BERT's representation for "[CLS] + c + [SEP] + q + [SEP]" or "[CLS] + q + [SEP]", whichever is relevant.

[4]The code for these experiments can be found under https://github.com/SgfdDttt/sara

[5]For a company, a goal would be to have 100% accuracy (resulting in no tax penalties) while paying the lowest amount of taxes possible (giving them something of an interest-free loan, even if the IRS eventually collects the understated tax).

Model	Features	Inputs	Entailment	Numerical
Baseline	-	-	50	20 ± 15.8
BERT-based	BERT	question	48	20 ± 15.8
		context	55	15 ± 14.1
		statutes	49	5 ± 8.6
	+ unfreeze	statutes	53	20 ± 15.8
	Legal BERT	question	48	5 ± 8.6
		context	49	5 ± 8.6
		statutes	48	5 ± 8.6
	+ unfreeze	statutes	49	15 ± 14.1
feedforward neural	tax vectors	question	54	20 ± 15.8
		context	49	20 ± 15.8
		statutes	50	20 ± 15.8
	word2vec	question	50	20 ± 15.8
		context	50	20 ± 15.8
		statutes	50	20 ± 15.8
feedforward non-neural	tax vectors	question	51	20 ± 15.8
		context	51	20 ± 15.8
		statutes	47	25 ± 17.1
	word2vec	question	52	20 ± 15.8
		context	51	20 ± 15.8
		statutes	53	20 ± 15.8

Table 5: Test set scores. We report the 90% confidence interval. All confidence intervals for entailment round to 8.3%.

5.2 Feedforward models

We follow Arora et al. [2] to embed strings into vectors, with smoothing parameter equal to 10^{-3}. We use either tax vectors described in Section 4 or word2vec vectors [39]. We estimate unigram counts from the corpus used to build the tax vectors, or the training set, whichever is relevant. For a given context c and question or prompt q, we retrieve relevant subsection s as above. Using Arora et al. [2], s is mapped to vector v_s, and (c, q) to v_{c+q}. Let $r = [v_s, v_{q+c}, |v_s - v_{c+q}|, v_s \odot v_{c+q}]$ where $[a, b]$ is the concatenation of a and b, $|.|$ is the element-wise absolute value, and $\odot$ is the element-wise product. The answer is predicted as $g(\theta_1 \cdot f(r))$ or $\mu + \sigma\theta_2 \cdot f(r)$, as above, where f is a feed-forward neural network. We use batch normalization between each layer of the neural network [31]. As above, we perform ablation experiments, where we drop the statute, or the context and the statute, in which case r is replaced by v_{c+q} or v_q. We also experiment with f being the identity function (no neural network). Training is otherwise done as above, but without the warmup schedule.

5.3 Results

We report the accuracy on the test set (in %) in Table 5. In our ablation experiments, "question" models have access to the question only, "context" to the context and question, and "statute" to the statutes, context and question. For entailment, we use a majority baseline. For the numerical questions, we find the constant that minimizes the hinge loss on the training set up to 2 digits: $11,023. As a check, we swapped in the concatenation of the RTE datasets of Bentivogli et al. [5], Dagan et al. [16], Giampiccolo et al. [26], Haim et al. [28], and achieved 73.6% accuracy on the dev set with BERT, close to numbers reported in Wang et al. [59]. BERT was trained on Wikipedia, which contains snippets of law text: see article *United States Code* and links therefrom, especially *Internal Revenue Code*. Overall, models perform comparably to the baseline, independent of the underlying method. Performance remains mostly unchanged when dropping the statutes or statutes and context, meaning that models are not utilizing the statutes. Adapting BERT or word vectors to the legal domain has no noticeable effect. Our results suggest that performance will not be improved through straightforward application of a large-scale language model, unlike it is on other datasets: Raffel et al. [45] achieved 94.8% accuracy on COPA [49] using a large-scale multitask Transformer model, and BERT provided a huge jump in performance on both SQuAD 2.0 [46] (+8.2 F1) and SWAG [63] (+27.1 percentage points accuracy) datasets as compared to predecessor models, pre-trained on smaller datasets.

Here, we focus on the creation of resources adapted to the legal domain, and on testing off-the-shelf and historical solutions. Future work will consider specialized reasoning models.

6 RELATED WORK

There have been several efforts to translate law statutes into expert systems. Oracle Policy Automation has been used to formalize rules in a variety of contexts. TAXMAN [38] focuses on corporate reorganization law, and is able to classify a case into three different legal types of reorganization, following a theorem-proving approach. Sergot et al. [52] translate the major part of the British Nationality Act 1981 into around 150 rules in micro-Prolog, proving the suitability of Prolog logic to express and apply legislation. Bench-Capon et al. [4] further discuss knowledge representation issues. Closest to our work is Sherman [53], who manually translated part of Canada's Income Tax Act into a Prolog program. To our knowledge, the projects cited did not include a dataset or task that the programs were applied to. Other works have similarly described the formalization of law statutes into rule-based systems [24, 30, 32, 51].

Yoshioka et al. [62] introduce a dataset of Japanese statute law and its English translation, together with questions collected from the Japanese bar exam. To tackle these two tasks, Kim et al. [33] investigate heuristic-based and machine learning-based methods. A similar dataset based on the Chinese bar exam was released by Zhong et al. [66]. Many papers explore case-based reasoning for law, with expert systems [43, 56], human annotations [8] or automatic annotations [3] as well as transformer-based methods [44]. Some datasets are concerned with very specific tasks, as in tagging in contracts [10], classifying clauses [11], and classification of documents [12] or single paragraphs [6]. Ravichander et al. [47] have released a dataset of questions about privacy policies, elicited from turkers and answered by legal experts. Saeidi et al. [50] frame the task of statutory reasoning as a dialog between a user and a dialog agent. A single rule, with or without context, and a series of followup questions are needed to answer the original question. Contrary to our dataset, rules are isolated from the rest of the body of rules, and followup questions are part of the task.

Clark et al. [14] describe a decades-long effort to answer science exam questions stated in natural language, based on descriptive knowledge stated in natural language. Their system relies on a variety of NLP and specialized reasoning techniques, with their most significant gains recently achieved via contextual language modeling. This line of work is the most related in spirit to where we believe research in statutory reasoning should focus. An interesting contrast is that while scientific reasoning is based on understanding the physical world, which in theory can be informed by all manner of evidence beyond texts, legal reasoning is governed by human-made rules. The latter are true by virtue of being written down and agreed to, and are not discovered through evidence and a scientific process. Thus, statutory reasoning is an exceptionally pure instance of a reasoner needing to understand prescriptive language.

Weston et al. [60] introduced a set of *prerequisite toy tasks* for AI systems, which require some amount of reasoning and common sense knowledge. Contrary to the present work, the types of question in the train and test sets are highly related, and the vocabulary overlap is quite high. Numeric reasoning appears in a variety of MR challenges, such as in DROP [19].

Understanding procedural language – knowledge needed to perform a task – is related to the problem of understanding statutes, and so we provide a brief description of some example investigations in that area. Zhang et al. [65] published a dataset of how-to instructions, with human annotations defining key attributes (actee, purpose...) and models to automatically extract the attributes. Similarly, Chowdhury et al. [13] describe a dataset of human-elicited procedural knowledge, and Wambsganß and Fromm [58] automatically detect repair instructions from posts on an automotive forum. Branavan et al. [7] employed text from an instruction manual to improve the performance of a game-playing agent.

7 CONCLUSION

We introduce a resource of law statutes, a dataset of hand-curated rules and cases in natural language, and a symbolic solver able to represent these rules and solve the challenge task. Our hand-built solver contrasts with our baselines based on current NLP approaches, even when we adapt them to the legal domain.

The intersection between NLP and the legal domain is a growing area of research [3, 11, 33, 35, 48], but with few large-scale systematic resources. Thus, in addition to the exciting challenge posed by statutory reasoning, we also intend this paper to be a contribution to legal-domain natural language processing.

Given the poor out-of-the box performance of otherwise very powerful models, this dataset, which is quite small compared to typical MR resources, raises the question of what the most promising direction of research would be. An important feature of statutory reasoning is the relative difficulty and expense in generating carefully constructed training data: legal texts are written for and by lawyers, who are cost-prohibitive to employ in bulk. This is unlike most instances of MR where everyday texts can be annotated through crowdsourcing services. There are at least three strategies open to the community: automatic extraction of knowledge graphs from text with the same accuracy as we did for our Prolog solver [57]; improvements in MR to be significantly more data efficient in training; or new mechanisms for the efficient creation of training data based on pre-existing legal cases.

Going forward, we hope our resource provides both (1) a benchmark for a challenging aspect of natural legal language processing as well as for machine reasoning, and (2) legal-domain NLP models useful for the research community.

REFERENCES

[1] 2019. Caselaw Access Project. http://case.law

[2] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings. (2016).

[3] Kevin D Ashley and Stefanie Brüninghaus. 2009. Automatically classifying case texts and predicting outcomes. *Artificial Intelligence and Law* 17, 2 (2009), 125–165.

[4] Trevor JM Bench-Capon, Gwen O Robinson, Tom W Routen, and Marek J Sergot. 1987. Logic programming for large scale applications in law: A formalisation of supplementary benefit legislation. In *Proceedings of the 1st international conference on Artificial intelligence and law*. 190–198.

[5] Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The Fifth PASCAL Recognizing Textual Entailment Challenge.. In *TAC*.

[6] Carlo Biagioli, Enrico Francesconi, Andrea Passerini, Simonetta Montemagni, and Claudia Soria. 2005. Automatic semantics extraction in law documents. In *Proceedings of the 10th international conference on Artificial intelligence and law*. 133–140.

[7] SRK Branavan, David Silver, and Regina Barzilay. 2012. Learning to win by reading manuals in a monte-carlo framework. *Journal of Artificial Intelligence Research* 43 (2012), 661–704.

[8] Stefanie Bruninghaus and Kevin D Ashley. 2003. Predicting outcomes of case based legal arguments. In *Proceedings of the 9th international conference on Artificial intelligence and law*. 233–242.

[9] Hector Neri Castañeda. 1967. Comment on D. Davidson's "The logical forms of action sentences". *The Logic of Decision and Action* (1967).

[10] Ilias Chalkidis and Ion Androutsopoulos. 2017. A Deep Learning Approach to Contract Element Extraction.. In *JURIX*. 155–164.

[11] Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2018. Obligation and prohibition extraction using hierarchical rnns. *arXiv preprint arXiv:1805.03871* (2018).

[12] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Large-Scale Multi-Label Text Classification on EU Legislation. *arXiv preprint arXiv:1906.02192* (2019).

[13] Debajyoti Paul Chowdhury, Arghya Biswas, Tomasz Sosnowski, and Kristina Yordanova. 2020. Towards Evaluating Plan Generation Approaches with Instructional Texts. *arXiv preprint arXiv:2001.04186* (2020).

[14] Peter Clark, Oren Etzioni, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, Sumithra Bhakthavatsalam, et al. 2019. From'F'to'A'on the NY Regents Science Exams: An Overview of the Aristo Project. *arXiv preprint arXiv:1909.01958* (2019).

[15] Robin Cooper, Dick Crouch, Jan Van Eijck, Chris Fox, Johan Van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, et al. 1996. *Using the framework.* Technical Report. Technical Report LRE 62-051 D-16, The FraCaS Consortium.

[16] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. Springer, 177–190.

[17] Donald Davidson. 1967. The logical forms of action sentences. *The Logic of Decision and Action* (1967).

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

[19] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161* (2019).

[20] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008. Open information extraction from the web. *Commun. ACM* 51, 12 (2008), 68–74.

[21] Edward A Feigenbaum. 1992. Expert systems: principles and practice. (1992).

[22] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building Watson: An overview of the DeepQA project. *AI magazine* 31, 3 (2010), 59–79.

[23] Noah S Friedland, Paul G Allen, Gavin Matthews, Michael Witbrock, David Baxter, Jon Curtis, Blake Shepard, Pierluigi Miraglia, Jurgen Angele, Steffen Staab, et al. 2004. Project halo: Towards a digital aristotle. *AI magazine* 25, 4 (2004), 29–29.

[24] Wachara Fungwacharakorn and Ken Satoh. 2018. Legal Debugging in Propositional Legal Representation. In *JSAI International Symposium on Artificial Intelligence*. Springer, 146–159.

[25] Bryan A Gardner. 2019. *Black's Law Dictionary* (11 ed.).

[26] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*. Association for Computational Linguistics, 1–9.

[27] David Gunning, Vinay K Chaudhri, Peter E Clark, Ken Barker, Shaw-Yi Chaw, Mark Greaves, Benjamin Grosof, Alice Leung, David D McDonald, Sunil Mishra, et al. 2010. Project Halo Update—Progress Toward Digital Aristotle. *AI Magazine* 31, 3 (2010), 33–58.

[28] R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

[29] Herbert Lionel Adolphus Hart and Herbert Lionel Adolphus Hart. 2012. *The concept of law*. Oxford university press.

[30] Robert Hellawell. 1980. A computer program for legal planning and analysis: Taxation of stock redemptions. *Columbia Law Review* 80, 7 (1980), 1363–1398.

[31] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[32] Imran Khan, Muhammad Sher, Javed I Khan, Syed M Saqlain, Anwar Ghani, Husnain A Naqvi, and Muhammad Usman Ashraf. 2016. Conversion of legal text to a logical rules set from medical law using the medical relational model and the world rule model for a medical decision support system. In *Informatics*, Vol. 3. Multidisciplinary Digital Publishing Institute, 2.

[33] Mi-Young Kim, Juliano Rabelo, and Randy Goebel. 2019. Statute Law Information Retrieval and Entailment. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*. 283–289.

[34] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[35] Anastassia Kornilova and Vladimir Eidelman. 2019. BillSum: A Corpus for Automatic Summarization of US Legislation. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. Association for Computational Linguistics, Hong Kong, China, 48–56. https://doi.org/10.18653/v1/D19-5406

[36] John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).

[37] Robert S Ledley and Lee B Lusted. 1959. Reasoning foundations of medical diagnosis. *Science* 130, 3366 (1959), 9–21.

[38] L Thorne McCarty. 1976. Reflections on TAXMAN: An experiment in artificial intelligence and legal reasoning. *Harv. L. Rev.* 90 (1976), 837.

[39] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[40] Randolph A Miller, Harry E Pople Jr, and Jack D Myers. 1982. Internist-I, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine* 307, 8 (1982), 468–476.

[41] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhanava Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Commun. ACM* 61, 5 (2018), 103–115.

[42] Terence Parsons. 1990. *Events in the Semantics of English*. Vol. 334. MIT press Cambridge, MA.

[43] Walter G Popp and Bernhard Schlink. 1974. Judith, a computer program to advise lawyers in reasoning a case. *Jurimetrics J.* 15 (1974), 303.

[44] Juliano Rabelo, Mi-Young Kim, and Randy Goebel. 2019. Combining Similarity and Transformer Methods for Case Law Entailment. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*. 290–296.

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[46] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

[47] Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. 2019. Question Answering for Privacy Policies: Combining Computational and Legal Perspectives. *arXiv preprint arXiv:1911.00841* (2019).

[48] Edwina L Rissland, Kevin D Ashley, and Ronald Prescott Loui. 2003. AI and Law: A fruitful synergy. *Artificial Intelligence* 150, 1-2 (2003), 1–15.

[49] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

[50] Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading. *arXiv preprint arXiv:1809.01494* (2018).

[51] Ken Satoh, Kento Asai, Takamune Kogawa, Masahiro Kubota, Megumi Nakamura, Yoshiaki Nishigai, Kei Shirakawa, and Chiaki Takano. 2010. PROLEG: an implementation of the presupposed ultimate fact theory of Japanese civil code by PROLOG technology. In *JSAI International Symposium on Artificial Intelligence*. Springer, 153–164.

[52] Marek J. Sergot, Fariba Sadri, Robert A. Kowalski, Frank Kriwaczek, Peter Hammond, and H Terese Cory. 1986. The British Nationality Act as a logic program. *Commun. ACM* 29, 5 (1986), 370–386.

[53] David M Sherman. 1987. A Prolog model of the income tax act of Canada. In *Proceedings of the 1st international conference on Artificial intelligence and law*. 127–136.

[54] Edward H Shortliffe and Bruce G Buchanan. 1975. A model of inexact reasoning in medicine. *Mathematical biosciences* 23, 3-4 (1975), 351–379.

[55] Jerrold Soh, How Khang Lim, and Ian Ernst Chai. 2019. Legal Area Classification: A Comparative Study of Text Classifiers on Singapore Supreme Court Judgments. Association for Computational Linguistics, Minneapolis, Minnesota.

[56] Anne vdL Gardner. 1983. The design of a legal analysis program. In *AAAI-83*. 114–118.

[57] Lai Dac Viet, Vu Trong Sinh, Nguyen Le Minh, and Ken Satoh. 2017. ConvAMR: Abstract meaning representation parsing for legal document. *arXiv preprint arXiv:1711.06141* (2017).

[58] Thiemo Wambsganß and Hansjörg Fromm. 2019. Mining User-Generated Repair Instructions from Automotive Web Communities. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*.

[59] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*. 3261–3275.

[60] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698* (2015).

[61] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2369–2380. https://doi.org/10.18653/v1/D18-1259

[62] Masaharu Yoshioka, Yoshinobu Kano, Naoki Kiyota, and Ken Satoh. 2018. Overview of japanese statute law retrieval and entailment task at coliee-2018. In *Twelfth International Workshop on Juris-informatics (JURISIN 2018)*.

[63] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326* (2018).

[64] Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Broad-Coverage Semantic Parsing as Transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3786–3798. https://doi.org/10.18653/v1/D19-1392

[65] Ziqi Zhang, Philip Webster, Victoria S Uren, Andrea Varga, and Fabio Ciravegna. 2012. Automatically Extracting Procedural Knowledge from Instructional Texts using Natural Language Processing.. In *LREC*, Vol. 2012. 520–527.

[66] Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2019. JEC-QA: A Legal-Domain Question Answering Dataset. *arXiv preprint arXiv:1911.12011* (2019).