BillSum: A Corpus for Automatic Summarization of US Legislation


## Abstract

We introduce BillSum, the first dataset for summarization of US Congressional and California state bills. Then, we benchmark extractive methods that consider neural sentence representations and traditional contextual features. Finally, we demonstrate that models built on Congressional bills can be used to summarize California bills, thus, showing that methods developed on this dataset can transfer to states without human-written summaries.


## 1. Introduction

We introduce the BillSum dataset, which contains a primary corpus of 22218 US Congressional bills and reference summaries split into a train and a test set.

Since the motivation for this task is to apply models to new legislatures, the corpus contains an additional test set of 1237 California bills and reference summaries.

## 2. Background

The previous studies in this area either apply traditional domain-agnostic techniques or take advantage of the unique structures that are consistently present in legal proceedings (e.g precedent, law, background).

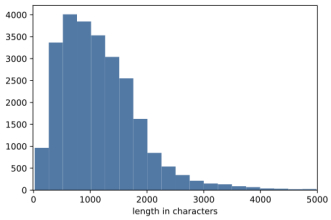
## 3. Data

The BillSum dataset consists of three parts: US training (18949 bills), US test (3269 bills) California test (1237 bills).

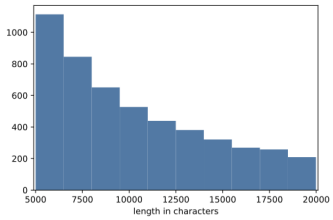
The BillSum corpus focuses on mid-length legislation from 5000 to 20000 characters in length. For the summaries, we chose a 2000 character limit as 90% of summaries are of this length or shorter.

The nested, bulleted structure is common to most bills, where each bullet can represent a sentence or a phrase.

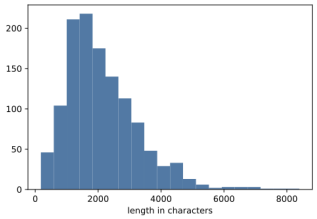
| | | mean | min | 25th | 50th | 75th | max |
|---|---|---|---|---|---|---|---|
| Words | US | 1382 | 245 | 923 | 1253 | 1758 | 8785 |
| | CA | 1684 | 561 | 1123 | 1498 | 2113 | 3795 |
| Sentences | US | 46 | 3 | 31 | 42 | 58 | 372 |
| | CA | 47 | 12 | 31 | 42 | 59 | 137 |



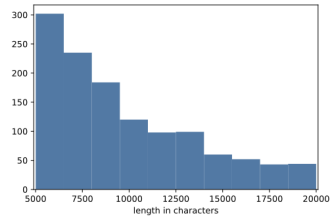

(a) US Bill Summaries




(b) US Bill Text




(c) CA Bill Summaries




(d) CA Bill Text

## 4. Benchmark Methods

We evaluate several extractive summarization approaches by first scoring individual sentences, then using a selection strategy to pick the best subset.

The scoring task is framed as a supervised learning problem. First, we create a binary label for each sentence indicating whether it belongs in the summary. We compute a ROUGE-2 precision score of a sentence relative to the reference summary and simplify it to a binary value by applying a threshold of 0.1.

Second, we build several models to predict the label. We consider two aspects of a sentence: its importance in the context of the document (4.1) and its general summary-like properties (4.2).

### 4.1. Document Context Model (DOC)

We encode the position of sentence as a fraction of ”sentence position / total sentence count” to restrict this feature to the 0 - 1 range (feature 1). In addition, we include a binary feature for whether the sentence is near a section header (feature 2).

We calculate a document-level TF-IDF weight for each word, then take the average and the maximum of these weights for a sentence as features (feature 3, 4). To relate language between sentences, we calculate sentence-level TF-IDF using each sentence as a document for the background corpus, the average and max of the sentence’s word weights are used as featured (feature 5, 6).

We train a random forest ensemble model over these features with 50 estimators. This method was chosen because it best captured the interactions between the small number of features.

### 4.2. Summary Language Model (SUM)

We pretrained the Bert-Large Uncased model on the "next sentence prediction" task using the US training dataset for 20000 steps with a batch size of 32.

Using the pretrained model, the classification setup for BERT is trained on sentences and binary labels for 3 epochs over the training data.

### 4.3. Ensemble and Sentence Selection

To combine the signals from DOC and SUM models, we create an ensemble averaging the two probability outputs. We apply MRR algorithm (Maximal Marginal Relevance).

$$s_{next} = \max_{s \in D - S_{cur}} 0.7 * f(s) - 0.3 * sim(s, S_{cur})$$

where $D$ is the document, $S_{cur}$ are the sentences in the summary so far, $f(s)$ is the output of DOC (1D vector), $s$ in $sim(s, S_{cur})$ is the output of SUM (2D vector). We repeat this process until we reach the length limit of 2000 characters.

## 5. Results

We evaluated the DOC, SUM, and ensemble classifiers separately. All three of our models outperform the other baselines. The SUM outperforms the DOC. The DOC+SUM improved the performance because of incorporating several contextual features.

| | Rouge-1 | Rouge-2 | Rouge-L |
|---|---|---|---|
| Oracle | 45.11 | 28.74 | 37.38 |
| SumBasic | 30.74 | 14.16 | 23.92 |
| LSA | 32.64 | 15.69 | 26.26 |
| TextRank | 34.35 | 17.77 | 27.80 |
| DOC | 38.51 | 21.38 | 31.49 |
| SUM | 40.69 | 23.88 | 33.65 |
| DOC + SUM | 40.80 | 23.83 | 33.73 |

| | Rouge-1 | Rouge-2 | Rouge-L |
|---|---|---|---|
| Oracle | 48.61 | 32.83 | 41.94 |
| SumBasic | 35.47 | 16.18 | 29.98 |
| LSA | 35.06 | 16.34 | 29.93 |
| TextRank | 35.81 | 18.10 | 29.97 |
| DOC | 38.35 | 19.76 | 32.80 |
| SUM | 38.90 | 20.79 | 33.20 |
| DOC + SUM | 39.65 | 21.14 | 34.05 |

Figure 1: (a) Congressional Bills (b) CA Bills

5 June, 2023