Yea or Nay: A Novel Approach to Understanding Congressional Debates

Lucas Pauker
Stanford University
Computer Science
lpauker@stanford.edu

Nina Prabhu
Stanford University
Computer Science
nprabhu@stanford.edu

Gilbert Rosal
Stanford University
Computer Science
grosal@stanford.edu

Abstract

Understanding the massive volumes of transcripts that come out of congress is often difficult and inaccessible. We propose an approach to congressional representative stance detection on the ConVote dataset, a dataset introduced by Thomas et al. (2006), to predict how representatives will vote based on their speeches in congress. We use past voting behavior as features as well as a distilBERT model finetuned on ConVote training data to perform this analysis. We acheive an overall test data set accuracy of 79.2% and F1-score of 80.0%, outperforming the baseline SVM and neural network models by around 10%.

1 Introduction

The US Congress releases transcripts of every congressional session. However, these transcripts are often difficult and time consuming to parse and understand due to their length and the jargon used by politicians. As a result, it is difficult for journalists and researchers to understand congressional proceedings. However, understanding congressional sessions is valuable since it provides an insight into how democracy works in the US. Furthermore, it is good for people to be able to understand what happens in congress quickly and effectively. Although politicians may make promises during campaigning, their true colors show on the congress floor, where they are faced with making actual decisions via voting on legislature. In this paper, we use NLU algorithms to parse congressional transcripts and analyze the sentiment of the politicians in Congress on various topics to predict how they will vote on these topics.

We use a dataset that includes debate text and how legislators voted for all bills during the 109th United States Congress in 2005 (Thomas et al., 2006). We predict how legislators voted on bills using textual data from these debates. We implement state-of-the-art BERT models and create new features to build a model that vastly outperforms our baseline.

2 Related Work

Past work in this area has consisted of a few approaches. We will highlight the most successful ones.

2.1 Sentiment Analysis

"Get out the vote: Determining support or opposition from Congressional floor-debate transcripts" (Thomas et al., 2006) performed sentiment analysis on individual congress peoples speeches to determine whether they were for or against a given topic. Unlike previous methods, the authors leveraged grounding relative to other speakers in the chamber to better understand the given speakers feelings, trying to interpret phrases like "I second that!" This is especially useful because congressional speeches are varied in topic and latent with misdirection and contextual phrasing. This paper highlights another difficult challenge that arises in general congressional sentiment analysis in its discussion about evidence. A given congressperson may respond to another's points or bring up evidence without saying "I agree with the legislation!" which means we need to also factor in the context from the bill itself. Overall, the paper employs an SVM that optimizes its labeling of speech segments based on both individual speech-segment classification scores and preferences for groups of speech segments to receive the same label.

Furthermore, this group found a significant improvement in prediction of the sentiment of speeches when the different speeches made by a given speaker were constrained to receive the same label. This is reasonable since we generally expect the same speaker to not drastically change their opinion throughout a congressional session (although it certainly may happen). However, we relaxed this constraint during some of our experiments to see if we could achieve similar results through more thorough sentiment analysis instead of manual constraints.

Thomas et al.'s (2006) success with sentiment analysis informed our methods, but we decided to focus more on similarities between bills instead of similarities between congresspeople in our final approach.

2.2 Contextual Feature Extraction

In "Predicting the Vote using Legislative Speech" (Budhwar, 2018), Budhwar used deliberation transcripts to form models of voting behavior for each legislator involved and make predictions of their voting activity. To make a comprehensive model, he extracted features such as number of questions asked and number of interruptions by each legislator, in addition to more standard features such as number of words spoken and sentiment of each utterance. The model used various natural language processing and sentiment analysis modules to process the text, as well as SVMs to find the difference between the Republican and Democrat votes.

By extracting features that relate legislators to each other instead of just analyzing legislator-specific utterances, this paper was able to achieve high accuracy and get the most out of contextual information. The feature extraction section of the paper is likely the most valuable to us, as it differentiates this task from many other NLU tasks that simply analyze one section of text at a time, and addresses and validates the possibility of gleaning information about legislators from their short interactions with other legislators.

Loosely inspired by this paper, we performed feature extraction to gain more information about the strength of legislator's feelings on different topics. We noted exclamation points as a sign of particularly high enthusiasm, and question marks as medium enthusiasm, as interest is being shown in a topic. This more accurately informed the strength of certain sentiments.

Budhwar also noted that there are newer NLP models that could improve upon the work done on his paper. As BERT is one of the newer methods and relatively unused in this task, we decided to use it as one of our components to see its success and develop richer representations of deliberation text.

2.3 Neural Network Approach

Many papers related to our task use legislators' past voting history to inform their future votes. However, some legislators have very little voting history, making it hard to predict what they will do next. In "Roll Call Vote Prediction with Knowledge Augmented Models", (Patil et al., 2019) Patil et al., the authors address this problem by finding other sources such as news text, and using these sources to further inform past voting history.

By making unigram features from news sources and a knowledge base embedding, they constructed a neural network to predict voting decisions of legislators on bills that outperformed current leaders. They represented each news text source that mentions a politician as a bag of words model of the most frequent 2000 words for articles across all politicians. They also used the Freebase Knowledge base to create another model. For this, they found all relationships in the KB that contained the politician, and passed these relationship triples as input into their model. They used both of these models to provide further input (additional information) models that take in the voting record of a politician, the politician, and a bill to predict whether the politician will vote yes or no on the bill.

We used neural networks in our early experimentation with our own features because of their superior performance with feature extraction in this paper. We chose to not use unigrams as one of our extracted features in order to explore other methods, as we knew from this paper that unigrams would be successful.

Patil et al. (2019) noted that BERT is an avenue for future work due to its novelty and ability to produce richer representations for bill text. In our project, we used BERT to produce representations for legislative text, as this has not been done in the papers we reviewed on our task.

2.4 Embeddings

Kraft et al. (2016) developed a model that uses pre-trained word embeddings to determine which words in a bill's text have the most bearing on voting activity. With these word embeddings, they trained multidimensional vectors for each congressperson and bill as an extension of the single-dimensional ideal point model.

	Total	Train	Test	Development
Speech segments	3857	2740	860	257
Debates	53	38	10	5
Average number of speech segments per debate	72.8	72.1	86.0	51.4
Average number of speakers per debate	32.1	30.9	41.1	22.6

Table 1: Corpus statistics for our data set. (Thomas et al., 2006)

Their model took in a congressperson and a set of unique words in a bill, and output a prediction of how the congressperson voted on the bill (yes or no).

While the set of unique words in a bill clearly influences voting behavior, we believed that reducing this to a set might reduce the importance of a word that is mentioned multiple times for emphasis. While this may not be a huge limitation in a formal bill, it is relevant for legislative debate transcripts, where people speak with intention and conversation is more informal and natural. Although this paper was successful, there is a stark difference between the texts we are analyzing (bill text versus legislative debate transcripts), so we used all of the words in a bill instead of just the unique ones.

3 Data

We used the data set from Thomas et al. (2006) for our analysis. This data set consists of all US floor debates in the House of Representatives during 2005 (3268 pages of transcripts in total). This data set is particularly useful since it contains annotated information for each discourse about the speaker, the bill being discussed, and whether the speaker voted for or against the bill being discussed, making it easy to start our experimentation.

The main data set Thomas et al. (2006) used was generated through the following steps:

1. Download all the 2005 House debates from `govtrack.us`.

2. For each page of debate, count the references to each bill then associate the page with the bill with the most references. If no references are made to a bill during the debate, discard the page.

3. Parse each page of the debate into speech segments, associating each speech segment with a specific speaker.

4. Download files from `govtrack.us` containing the votes placed by each representative for the each bill throughout 2005. Connect the debate transcripts to this data so that we have information about whether each speech segment speaker supports or opposes the bill being discussed. If a speaker abstained from a vote, that speech segment is discarded.

5. Group sets of speeches corresponding to the same bill into a ”debate”. Debates with less than 20% of speech segments in favor of the bill and opposed to the bill are discarded for too one-sided.

Table 1 includes a breakdown of key statistics about the dataset. The train/test/development split used in our analysis is the same as used in Thomas et al. (2006).

This dat aset is also convenient since there are many other papers in the NLU literature that have used this data. Such papers are described at this webpage: `https://www.cs.cornell.edu/home/llee/data/convote.html`.

Another important note: 80% of bills were introduced by republicans in 2005, the time period that this dataset comes from. In 2005, the GOP controlled both the house and senate, explaining the reasoning for a trend we notice later: congress people will vote in similar ways on similar bills.

4 Baseline

We used two primary baselines that we compared our novel models to.

1. A baseline that uses an SVM classifier to classify each speech segment using a vector-

ized representation of the speech segment and reweighted using TF-IDF.

2. A baseline that uses a neural network classifier to classify each speech segment using a vectorized representation of the speech segment and reweighted using TF-IDF. We optimized this neural network using a grid search and found ended up using a model with three hidden layers. The hidden layers had 50, 10, and 2 nodes, respectively.

5 Metrics

Our main goal in this project will be to use textual data to predict how representatives in Congress will vote on an issue. Therefore, we used F1-score and accuracy as our primary metrics. The accuracy is calculated as

$$\frac{\alpha}{\tau}, \tag{1}$$

where α is the number of correct predictions and τ is the total number of samples. A correct prediction is defined as when our model predicts the correct vote for a legislator based on the speeches a legislator made during a debate. The F1-score is defined as

$$2\,(\frac{1}{\pi}+\frac{1}{\rho})^{-1}, \tag{2}$$

which is the harmonic mean of the precision π and recall ρ. Precision and recall are defined as

$$\pi = \frac{\text{tp}}{\text{tp}+\text{fp}} \tag{3}$$

and

$$\rho = \frac{\text{tp}}{\text{tp}+\text{fn}}, \tag{4}$$

where tp is the number of true positives, fp is the number of false positives, and fn is the number of false negatives. The F1-score is a good metric since it balances precision and recall. A high F1-score as well as accuracy shows that our model is robust and generalizable.

6 Methods

For the task of senator stance prediction using textual data, we built two components: one leveraging manual feature extraction of past voting history and the other using BERT finetuned on the ConVote dataset (we referred to this model as gilBERT). We tested each of these models separately and ultimately combined the models. We will now discuss each model in detail.

6.1 Bill Similarity

We extended traditional supervised learning models with novel features. Specifically, we focused on implementing bill similarity and past voting history to improve our model. These two features use the following intuitive heuristic: people will vote in similar ways on similar bills. To implement bill similarity, we encoded text into a feature space then calculated the relative distance between documents using a distance function. We experimented with TF-IDF encodings as well as BERT encodings for the feature representation of the bill texts.

To calculate the similarity of two different bills, we first grouped all the debate text for each separate bill together. We then encoded each bill-specific text into a matrix using TF-IDF or BERT. Lastly, we calculate the dissimilarity of the text of two bills by using a distance function on the two vector representations of the bills. A higher distance implies that bills are more difference, while two bills that are exactly the same will be distance 0 from each other. Figure 1 shows the distributions of the distances between bills using TF-IDF encodings as well as BERT encodings. This figure also shows the distribution of how similar voting behavior is between different bills. We can see that both TF-IDF and BERT distances are qualitatively close to normal distributions, which shows that this statistic picks out interesting information that is distributed in an intuitive way.

BERT encodings are a relatively novel approach to this field. To extend our previous basic bill similarity computations, we decided to first each document to its respective BERT encoding using a phi function. This gave us a multidimensional vector for each document, with shape (x, 768) where x varied loosely related to the length of the document. To compare each encoding, we averaged each multidimensional vector along its first dimension, giving us a final shape of (1,768) or a row vector of length 768 that we could use in the method described in the previous paragraph.

Figure 2 is a scatter plot of the TF-IDF distance scores compared to the vote similarity. This plot shows that there is an inverse correlation between TF-IDF distance and vote similarity. This is expected since bills that are similar should have a low TF-IDF distance and a high voting similarity. This gives us some confidence that our hypothesis heuristic that people will vote on bills that have similar text has merit. However, this data is clearly

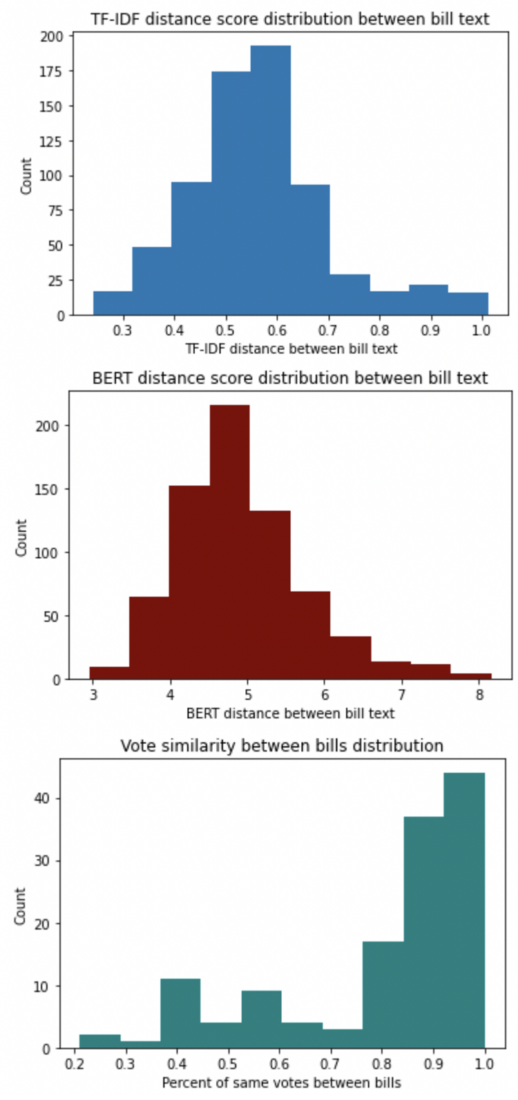

Figure 1: *Top*: Euclidean distance distribution between different bills encoded using a document-frequency matrix and reweighted using TF-IDF.
Middle: Euclidean distance distribution between different bills encoded using BERT.
Bottom: Distribution of how similar votes from the same representatives are between different bills. Similarity is calculated as a percentage of speakers in debates that voted the same way in the bills we are comparing.

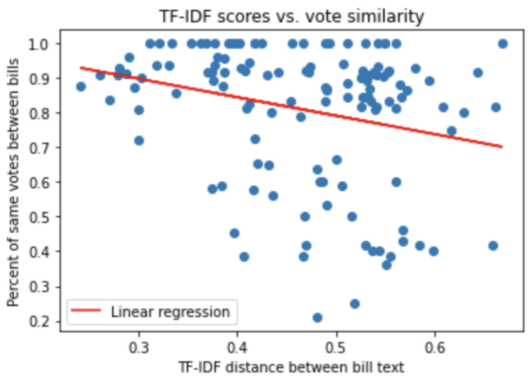

Figure 2: Graph of TF-IDF distance scores versus vote similarities. The red line is the line of best fit calculated using linear regression. We expect the slope to be negative since more similar bills (smaller TF-IDF distance) should result in representatives voting the same way on both bills. The line of best fit has a slope of -0.533, a y intercept of 1.058, and an R^2 coefficient of 0.066.

very noisy, as evidenced by the low R^2 statistic for our line of best fit. This is due to the fact that our voting similarity score only measures the voting similarity between representatives that spoke in the debate for both bills in question. Some debates having a small number of overall speakers. Therefore, many bills have few common speakers between them, the vote similarity score is a noisy statistic.

To make predictions using the distance calculated between bills, we use past voting behavior. For any bill that we seek to predict how a representative will vote, we first find the most similar bills in the training set to the bill in question by choosing the n bills with the lowest distance scores compared to the bill in question. We then look up how the representative voted on these similar votes, and output the majority way they voted. This introduces a hyperparameter n that represents the number of similar bills to consider. By using an exhaustive grid search, we found that $n = 26$ yields in the highest accuracy on the development data set. Results of our trials using the bill similarity classifier are shown in Table 3.

Furthermore, we have a choice of distance function that is used to calculate similarity between bills. We considered Euclidean and cosine distance functions. We found that Euclidean distance outperformed cosine distance in accuracy on our development data set. This makes sense, as (Gerrish and Blei, 2011) and other papers describe politician's viewpoints using the ideal point model. In this, politicians and bills can are mapped to points on a number line or coordinate plane and congresspeople vote on bills based on their distance to the bill. Taking this common representation into account, it follows that Euclidean distance would be better suited to this task than cosine distance.

6.2 BERT Model

Most work in congressional stance analysis is done using supervised machine learning models with explicit feature extraction. However, as noted by Budhwar (2018) these models are merely the foundation of future work as there are new deep learning NLP models that could significantly improve progress in this field, such as BERT (Budhwar, 2018). Patil et al. (2019) corroborate this claim as they explain that BERT could be a key because of its ability to represent text in a deeper, more meaningful way(Patil et al., 2019). As a result, we use a BERT model fine tuned on the Convote data set in an attempt to improve performance.

For our BERT model we used the following rationale as motivation: the utterances made by each congress person should be latent with enough meaning about their stance ("I hate this bill!") such that an attention-based model like BERT should be able to identify the stance of the speaker and make a corresponding prediction. We chose distilBERT because it was faster to train and fine tune on due to its smaller size. More specifically, distilBERT reduces the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster with 6-layers, 768-hiddens, 12-heads, and 66M parameters (Sanh et al., 2020).

The output of distilBERT is a prediction of yes or no given some speaker's utterance in a Congress debate, in line with of task of stance analysis. One major concern was that some of the utterances were more than 512 words (which is the cap for BERT), meaning we had to cut many of the inputs short. This means key information may have been truncated. If given more time and a more powerful computer, we also tried experimenting with other BERT models like bert-base-uncased, bert-large-uncased, and roBERTa but running all the tests proved too costly (with each test taking about 1.5 hours).

For the BERT model, we experimented with three primary hyperparameters: the number of gradient accumulation steps g, the number of epochs e, and whether the vocabulary was cased c. We experimented with gradient accumulation step values of 1 and 8, epochs of 3, 5, and 10, and both cased and uncased vocabulary. We ran our model on an AWS m5.8xlarge instance and used a grid search to determine the best parameters. We found that the best parameters were $g = 1$, $e = 10$, and $c =$ `False`. Immediately updating the gradient proved more effective because steps between gradients now had the previous examples gradient impacting the parameters. A lower gradient accumulation step size does, however, increase the amount of computation (since were updating the parameters 8x more often), so we have a tradeoff between compute resources and model performance. The BERT model tests on the development set with different hyperparameters can be seen in Table 2.

GAS	Epochs	Cased?	F1-score
8	3	Yes	0.698
8	3	No	0.703
1	3	Yes	0.733
8	5	No	0.737
1	3	No	0.739
1	5	No	0.746
1	10	No	0.748

Table 2: BERT macro F1-score for the development data set with varying hyperparameters. The rows are arranged in order of increasing F1-score. GAS refers to the number of gradiant accumulation steps.

6.3 Combination Model

To combine the BERT and bill similarity models, we used heuristics to determine whether to use the BERT prediction or the bill similarity prediction for a given speaker and debate text. Since our BERT model outputted the likelihood a specific speaker would vote yes or no, we used the BERT prediction if this likelihood was above a threshold accuracy. For this threshold, we simply used the accuracy of the bill similarity model with TF-IDF encodings on the test set, which is 0.789, as seen in Table 3. This threshold makes sense since if BERT is more confident than our bill similarity accuracy, we should use BERT instead of the bill similarity metric.

Model	Development accuracy	Development F1-score	Test accuracy	Test F1-score
Neural network baseline	0.646	0.668	0.636	0.669
SVM baseline	0.722	0.754	0.686	0.719
distilBERT model	0.735	0.748	0.721	0.729
Bill similarity model with TF-IDF encodings	0.897	0.907	0.789	0.798
Bill similarity model with BERT encodings	0.903	0.912	0.773	0.783
Combined model	0.903	0.913	0.792	0.800

Table 3: Comparison of different models discussed throughout the paper. We include accuracy and F1-scores for both the development and test data sets.

7 Results and Discussion

Table 3 contains the results from the test set. Our most successful model was the combined bill similarity and BERT model, followed by the bill similarity models, then the BERT model, then the baselines.

The success of the feature extracting models is likely attributable to the strong signal in a vote. By adding the past voting behavior, we are able to understand how a specific speaker voted and therefore are able to predict how they will vote on a similar new bill accurately.

The poorer performance from the BERT model stems from two main places. First, BERT is only compatible with utterances of max length 512, but this is problematic since many of our training examples had a length greater than 512. This means many of the training examples had to be truncated, potentially removing information essential to a good prediction. Additionally, although distilBERT can effectively learn the relationship between words within a sentence, it overlooks the relationship between documents since many of the utterances are responsive in nature.

We observed that the hybrid model performed better than any of its individual components with the highest test accuracy and F1-score of 0.792 and 0.800 respectively. This shows that there is a strong case for model hybridization for congressperson stance detection. That being said, Patil et al. (2019) were able to achieve scores in the 90% range using more outside context beyond congressional speakings, indicating much more room for future work.

Across every model, our development score was better than our test score in both F1 and accuracy statistics, likely indicating too much tuning on the validation set and thus overfitting. This makes sense in both the context of our bill similarity model and BERT model since both used the training set to continually fine tune their parameters (and for example, increasing the number of epochs for the BERT model).

The development set's results for both bill similarity models show significant promise for better predictions achieving a high of 90.3% when the combination model is used. The BERT model's performance paled in comparison to the other models reaching a 75% success rate, although this was still better than both the NN and SVM baselines.

8 Conclusion

Building off of Patil et al. (2019) and Budhwar (2018), this paper sought to derive latent meaning from politicians utterances in congressional chambers using both old and new techniques: manual feature extraction, BERT fine tuned on task-related datasets, and a combination of the two. All three methods did better than the baselines with the hybrid performing the best, followed closely by the manual feature extraction model, and all three models also generalized relatively well. This lays the groundwork for future experimentation with BERT and hybridization in this task space. Furthermore, this paper shows that considering past voting behavior as well as textual similarity between bills is essential for building an accurate model for predicting votes in Congress based on textual debate data.

9 Future Work

As an extension of this paper, one could further experiment with different BERT models (like bert-base or bert-large) and further tweak hyper parameters. Furthermore, one could find a better way to navigate the length of many data points being greater than 512. BERT itself shows significant promise in its ability to outperform baselines, and further investigation must occur to see its power in this field.

In this paper, we focused on predicting how congresspeople vote based on fixed bill text. However, there is also much work to be done with the converse of this problem - modeling the voting strategies of fixed congresspeople. Yano et al. (2012) sought to predict whether a bill will survive a congressional committee. They created models of how congress committee members would vote on certain bills and found that the text of bills contains important information that can help predict success in Congressional committees.

We can use our existing work in predicting how people will vote on known bills to take Yano et al.'s work further and investigate how voting behavior is affected by small changes in bill wording (e.g. replace one topical word with another, or flip all party words in a debate).

10 Acknowledgements

We would like to thank Professor Chris Potts, our advisor Dora Demsky, and the rest of the 224U course staff for teaching us information necessary to conduct the experiments described above and providing feedback on this project and other assignments along the way.

11 Authorship Statements

Nina worked on refining the BERT representations for the manual feature extraction model and create the neural network baseline model. She also helped with initial iterations of the gilBERT model. Lucas worked on the bill similarity model, implementing the logic for both the TF-IDF and BERT representations. Lucas also implemented the SVM baseline model. Gilbert worked on the gilBERT model, implementing the finetuning on ConVote, setting up the AWS machine, and performing any high volume experiments. All three of us worked together to implement the combined model and write all papers.

References

Aditya Budhwar. Predicting the vote using legislative speech. *null*, 03 2018.

Sean Gerrish and David Blei. Predicting legislative roll calls from text. Technical report, null, 2011. URL https://icml.cc/2011/papers/333_icmlpaper.pdf.

Peter Kraft, Hirsh Jain, and Alexander Rush. An embedding model for predicting roll-call votes. 11 2016.

Pallavi Patil, Kriti Myer, Ronak Zala, Arpit Singh, Sheshera Mysore, Andrew McCallum, Adrian Benton, and Amanda Stent. Roll call vote prediction with knowledge augmented models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 574–581, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1053. URL https://www.aclweb.org/anthology/K19-1053.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

Matt Thomas, Bo Pang, and Lillian Lee. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335, Sydney, Australia, July 2006. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W06-1639.

Tae Yano, Noah A. Smith, and John D. Wilkerson. Textual predictors of bill survival in congressional committees. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 793–802, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/N12-1097.