Phenom Detection for Summary Generation of Hearings

DANIEL CHING, California Polytechnic State University, San Luis Obispo, USA
SCOTT PRAMUK, California Polytechnic State University, San Luis Obispo, USA
LUKE FANGUNA, California Polytechnic State University, San Luis Obispo, USA

1 Problem, Background, and Context

In the United States, congressional hearings are meetings of a legislative body such as the Senate, House, Joint Committees, Special Committees, or state legislatures. These sessions seek to gather information and feedback on proposed legislation, conduct an investigation, or evaluate and oversee the activities of a government department. Hearings may be unrelated to legislation and instead focus on discussions regarding topics of current interest. These hearings are open to the public and transcripts of each hearing are readily available online. Our work aims to make progress toward summarizing these hearings by detecting phenoms in the transcripts. Phenom detection in the context of Natural Language Processing aims at capturing unusual phenomena in text. These phenomena can include mentions of "Named Entities" such as organizations; people; locations; etc., toxicity, sentiment and polarity, use of rhetorical devices, and the topics discussed. Our task is to identify such phenomena in political speech and present them to the end user in a readable form. We decided to define phenoms more narrowly as mentions of organizations, toxicity, and insights gained from analyzing sentiment.

Authors' Contact Information: Daniel Ching, dlching@calpoly.edu, California Polytechnic State University, San Luis Obispo, San Luis Obispo, California, USA; Scott Pramuk, smpramuk@calpoly.edu, California Polytechnic State University, San Luis Obispo, San Luis Obispo, California, USA; Luke Fanguna, lfanguna@calpoly.edu, California Polytechnic State University, San Luis Obispo, San Luis Obispo, California, USA.

2 Data Sources, Tools, and Services

2.1 Data Sources

Our primary data source is the "Digital Democracy" database. This database contains labeled data about bills, committees, hearings, discussions, and more from the California, Texas, Florida, and New York state legislatures. The data that we are primarily interested in is a table labeled 'Utterance', which contains human-transcribed speech data from legislative hearings. Some other data sources we used are the CoNLL 2003 [Tjong Kim Sang and De Meulder 2003] and the WNUT 2017 datasets [Derczynski et al. 2017] which were used when fine-tuning a generic bert-base-cased model for the task of Named Entity Recognition. We experimented with Bert-based models and a max entropy model to detect organizations.

2.2 Tools and Services

In order to organize and distribute our code to group members, we created a git repository on GitHub which can be found here. The project itself was implemented entirely in Python.

3 Software Design and Architecture

3.1 User Interface

We use IRC (Internet Relay Chat) to interact with users. We created a chatbot that accepts various commands. The first command is:[[botname]: list [integer]] which will output a list of all valid did (discussion id) values that begin with integer (ex: 102 may return [102,1021]). If no integer is provided, 40 random dids will be returned instead. Each did value is associated with one bill and a group of utterances that were said in legislative hearings concerning this bill. The second is [[botname]: show [did]]. If did is a valid did, this command will output a summary of the bill being discussed, a list of people and organizations that they mention (ex: Ken Vendor mentioned 'CPOA' 2 times), the most positive speaker and a summary of what they said, the most negative speaker and a summary of what they said, the main political topics discussed during the hearing, if any speaker dominated the discussion, and any levels of toxicity detected and the phrases associated with them. Not all phenoms will be present for each hearing. For example, in the case that a hearing has unlabeled utterances, the sentiment analysis phenoms will not be detected.

3.2 Sentiment Analysis Insights

Sentiment analysis is the process of grouping the attitudes expressed by a piece of text into categories that represent the general opinion of the author. For example, text may be classified as positive, negative, or neutral or into more specific groupings. Sentiment analysis can reveal insights into the speaker's opinion on certain topics or toward other people. For the purposes of this project, sentiment analysis was used to identify the most positive and negative speakers throughout the course of the discussion. Given that our corpus of text involves political speech, a sentiment analyzer trained specifically on political terminology would work best. To do so, we utilized *XLM-T-Sent-Politics* [Barbieri et al. 2022] which is Facebook's *xlm-roberta-base* [Conneau et al. 2019] model trained specifically on tweets from politicians. This model provides sentiment analysis scores for a given piece of text, categorizing text into Positive, Negative, or Neutral sentiment. Each of these categories has a probability associated with it indicating how strong the categorical association is with the inputted piece of text. To determine the most positive and negative speakers, the following algorithm was used to calculate the average sentiment of a speaker:

$$\frac{1}{m}\sum_{i=1}^{m}\left(\sum_{j=1}^{n_i}\left(\begin{cases} p_j & \text{if } l_j = \text{"Positive"} \\ -p_j & \text{if } l_j = \text{"Negative"} \end{cases}\right)\right) \tag{1}$$

where m is the total number of sentiment sets for each speaker, n_i is the number of labels in the i-th sentiment set, p_j is the probability of the j-th sentiment in the i-th sentiment set, and l_j is the label of the j-th sentiment in the i-th sentiment set. Note that Neutral sentiment is not used in the calculation. Speaker sentiment also tended to misrepresent speakers who spoke only once during the discussion, so only speakers who spoke more than once were considered. After identifying the most positive and negative speakers, summaries for each were generated to provide insight into their opinions. To do so, we utilized Mistral's *Mistral-Nemo-Instruct-2407* [AI and NVIDIA 2024] Large Language Model with a custom prompt to summarize the speaker's opinion:

Provide a complete and concise one or two sentence analysis of [speaker]'s opinion based only on their utterances: [utterances].

Multiple attempts were needed to fine-tune the prompt to limit made-up content not found in the speaker's utterances and ensure the generated text is not too long. The response is also filtered through a regex to ensure that only alphanumeric characters are returned.

3.3 Topic Identification

As another phenom, we wanted to identify some of the main political topics discussed during a hearing. The approach we took for doing so was first to create groups of words using Latent Dirichlet allocation and then use a Large Language Model to generate political headings based on the groupings. Latent Dirichlet allocation is a generative statistical model that automatically extracts topics from text. It works by first randomly assigning words to a preset k number of topics and then iterates over the words using probabilistic methods to refine the categorizations. For our case, we first tokenzied all utterances for the hearing of interest and removed stop words. We then defined k to be five and for the model to return the top fifteen words in each topic. Afterwards, we feed each group of words into the same LLM used in the Sentiment Analysis section with the following prompt:

Generate a political category for these words: [topic list]. Categories should be relevant and specific political issues. Respond only with a single category.

These categories are then filtered for duplicates and outputted as the main categories for the hearing.

3.4 Speaker Dominance Detection

Another phenom we wanted to detect was whether a single speaker participated significantly more in discussion compared to other speakers. To do so, we employed statistical methods for counting the proportion of words spoken for each speaker as well as the proportion of times the speaker spoke. These two proportions are then combined to form a dominance score according to the formula below:

$$\alpha * \frac{Words\ spoken}{Total\ words\ spoken} + \beta * \frac{Times\ spoken}{Total\ times\ spoken} \tag{2}$$

where $\alpha = 0.7$ and $\beta = 0.3$. These parameters were chosen to provide greater weight to speakers who speak more words less often over speakers who speak frequently but are brief.

3.5 Organization Detection

Organization detection in the context of this project consists of detecting the beginnings and inner parts of an organization, concatenating these constituent parts to form coherent organizations, and then validating the final result by comparing it to a known database of known organizations or checking for common keywords often associated with organizations. Initially, we create the tagged output which includes information about what text is tagged, the label associated with the text (e.g. B-ORG, I-PER):

```
def createTaggedOutputs(self, utterance):
    tagged_outputs = []
    ner_results = self.nlp(utterance)
    tagged_outputs.append(ner_results)
    return tagged_outputs
```

With this tagged output, we need to concatenate consecutive organizations detected. For instance, "Florida Hospital Association" may be split into three entries– "Florida" being tagged as B-ORG (beginning organization), "Hospital" being tagged as I-ORG (In organization), "Association" being tagged as I-ORG (In organization)– and we want to reconstruct the organization "Florida Hospital Association":

```
if entry['entity'] == 'B-ORG':
    # Start a new organization
    if current_org:
        current_org['word'] = self.clean_word
            (current_org['word'])
        merged_entities.append(current_org)
        current_org = ...
    elif entry['entity'] == 'I-ORG' and
        current_org:
        # Append to the current organization
        current_org['word'] += f" {entry['
            word']}"
        ...
    else:
        # Finalize the current organization
            if it exists
```

Finally, we attempt to validate the tagged organizations by comparing them to known organizations from the Digital Democracy database through a Levenstein distance. Through training we found 3 to be the best value for this distance. One drawback of using this relatively large value is that detected organizations with shorter names are very likely to match some known organizations.

```
#List of common words associated with
    organizations
org_keywords = ['LLC', 'Inc.', 'Corp.', '
    Limited', 'Ltd.', 'Co.', 'Foundation','
    Association', 'Institute', 'University',
    'College', 'Group', 'Enterprise', '
    Holding','Ventures', 'Partners']
if entity_type == "ORG":
    for csv_value in csv_values:
        if levenshtein_distance(text,
            csv_value) <= self.editDist:
            validated_orgs[text] = 1
            break  # Avoid duplicates if text
                matches multiple CSV values
        else:
            #For orgs not in DB, want to
                additionally check that they
                meet some keyword
                requirements
            if bool(re.search(pattern, text,
                re.IGNORECASE)):
                validated_orgs[text] = 1
```

We also want to detect organizations that are not in the known database. Since we are inherently less confident in these organizations as we cannot validate them against our database, we check that they contain one of the keywords commonly associated with organizations. This keyword list was created empirically after manually inspecting a subset of the data. The model also often classified single letters such as 'A' or 'T' as organizations. We decided to disregard any of these single-letter organizations as they were often times typos in the data or not actual organizations.

3.6 Toxicity Detection

Toxicity detection in a discussion was another phenom of interest. What we wanted to accomplish with this phenom is to find utterances in a discussion that were classified as Toxic. Utterances that were tagged as Toxic fell under seven categories: *Toxic, severe_toxic, obscene, threat, insult, identity_hate, sexual_explicit.* For example, if we had the sentence, "You have no idea what you're talking about", it would be tagged as toxic.

In order to begin to detect Toxicity, we had collect data and use a transformer. The data we used was the Utterances table in the Digital Democracy Database. This table has 100,000+ utterances in Bill Discussions and will help us understand how politicians speak and if we can gather any insight of how toxic they can be in discussions. The transformer we decided to use was Toxic-bert [Hanu and Unitary team 2020].

Toxic-bert is a bert-based model that has three versions: Unbiased, which is trained on Wikipedia Comments, Biased, which trained on Civil Comments, and Multilingual, which was trained on both Wikipedia and Civil Comments. We decided to use the Biased model because we realized that a biased model yielded results that we were more confident in. We used the model to label 65,000+ utterances. The labeling process would take the label of highest score and label the utterance if it reaches a certain threshold. The threshold we settled with was .1 because detecting toxicity was rare on its own. Although we have an interesting distribution of scores, Figure 1, these scores only represented <1% of the data.

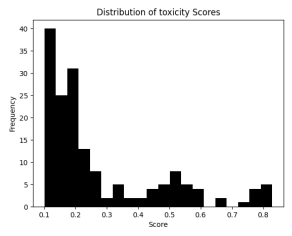

Fig. 1. Caption

Once we labeled the data, we began inspecting the labels and we noticed that out of the 65,000+ utterances, only 166 utterances were labeled as toxic. This made it difficult to evaluate and due to biases from the model, we couldn't effectively due an analysis. The limitations of the model came from not being able to detect intent of toxicity and would classify things as toxic if many words from the data appear in the text.

4 Evaluation Methodology

To evaluate our methods for detecting organizations, we decided to manually label a small subset of the utterances in the database. We first selected a random sample of 100,000 utterances. We used the bert-base-cased model fine-tuned on the CoNLL 2003 [Tjong Kim Sang and De Meulder 2003] to estimate the proportion of these utterances with organizations detected, which turned out to be around 15.2%. We then created two samples of 100 utterances each roughly matching this proportion and manually labeled each utterance with a list of organizations detected. We used the first set to tune a hyperparameter and the second set to test the precision, recall, and f1 scores of the models.

To evaluate the methods of detecting toxicity, we tested the unbiased and biased models against each other. The goal of this was to see how much toxicity the biased model could point out compared the the unbiased model. This was difficult because out of the 65,000+ utterances, only 166 were classified as toxic. When testing the accuracy, we did 10-fold cross-validation on a batch of 1000 randomly sampled utterances.

5 Evaluation Results

Using the labeled organization data, we evaluated 3 models: an NLTK-provided max entropy model, a bert-base-cased model fine-tuned on the CoNLL 2003 dataset [Tjong Kim Sang and De Meulder 2003], and this second model trained further on the WNUT 2017 dataset [Derczynski et al. 2017].

	Micro Precision	Micro Recall	Micro F1
NLTK	0.48	0.52	0.50
Bert 1	0.51	0.49	0.50
Bert 2	0.50	0.50	0.50

Table 1. Org Detection Evaluation

	Macro Precision	Macro Recall F1	Macro F1
NLTK	0.41	0.52	0.42
Bert 1	0.63	0.49	0.61
Bert 2	0.24	0.50	0.24

Table 2. Org Detection Evaluation

In evaluating the Toxicity detections, the accuracy turned out to be around 99.7%. This was uninteresting due to the fact that <1% of the data was actually labeled toxic.

6 Shortcomings and Challenges

We encountered multiple challenges while working on this project. For organization detection, our main hurdle was the nature of our dataset. Since many organizations, bills, programs, etc. are repetitively mentioned in these utterances, acronyms are often used. It can be very challenging, even for a human, to determine whether a certain acronym is referring to an organization, legal act, program, or something else entirely. Our model tended to classify many of these acronyms as organizations in situations where they were ambiguous, which led to some errors during testing. Another challenging part of the data is that names are often mentioned. This also creates ambiguity as names can be associated with companies (ex: Disney, Ford, etc.) and once again our model tended to classify many of these names as organizations incorrectly.

In using the LLMs to generate summaries and groupings for political topics, we found that prompts needed to be highly specific for the model to produce the desired output. For example, summaries would often include hallucinated information until we included in the prompt to generate the summary based only on the utterances provided. The model would also generate strange and random punctuation and spacing which needed to be corrected.

A challenge for toxicity detection was intent and the bias in the models. In some utterances that were labeled as toxic, it would not tell the difference between a personal experience and actual toxicity. It would identify a bunch of keywords and score it. Another challenge was intent. If it identified a bunch of keywords then it would automatically have a high score.

7 Conclusion and Future Work

Our research proposes three categories of phenoms in political speech: named organizations, sentiment and topics discussed, and toxicity. Our best organization detection model yielded 0.50 Micro F1 and 0.61 Macro F1 scores on the small labeled test data set. Future work on this facet of the project could include creating a larger labeled dataset from the Digital Democracy DB to be able to more robustly test predictions, and experimenting with other kinds of predictors such as LLMs, CRF-based models, and LSTM-CRF models. To expand on the sentiment analysis and topic identification explored here, these two ideas could be combined to identify a given speaker's sentiment towards different topics and track how it changes over time. It would also be more meaningful if the sentiment classification was divided into more specific categories rather than generic positive or negative. Topic identification could also be improved by running Latent Dirichlet allocation on the entire or a large subset of the Digital Democracy database to better identify categories of political topics discussed. For toxicity detection, we understood that the use of a transformer model is much more useful than a logic check. It detected more interesting utterances that we would not have found toxic. It would be important to generate more data to have more accurate predictions. Another future work for toxicity detection would be to figure out the intent of a statement instead of seeing a bunch of keywords and identifying it as toxic.

References

Mistral AI and NVIDIA. 2024. Mistral-Nemo-Instruct-2407. https://huggingface.co/mistralai/Mistral-Nemo-Instruct-2407 Available at HuggingFace repository.

Francesco Barbieri, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. XLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. European Language Resources Association, Marseille, France, 258–266. https://aclanthology.org/2022.lrec-1.27

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. *CoRR* abs/1911.02116 (2019). arXiv:1911.02116 http://arxiv.org/abs/1911.02116

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*. Association for Computational Linguistics, Copenhagen, Denmark, 140–147. https://doi.org/10.18653/v1/W17-4418

Laura Hanu and Unitary team. 2020. Detoxify. Github. https://github.com/unitaryai/detoxify.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 142–147. https://www.aclweb.org/anthology/W03-0419