

LingoQuartet

Advanced ML for Public Policy Final Project

JP Martinez, Jackie Glasheen, Lisette Solis, Megan Moore



Table of contents

- 01** Introduction & Overview
- 02** Data Overview
- 03** Methods: Clustering
- 04** Methods: Sentiment Analysis
- 05** Results
- 06** Future Work



Introduction

Goal: explore clustering techniques to identify topics covered in news articles and analyze sentiment and amount of coverage of popular topics.

Data Overview

- **512,978** articles in the initial data corpus
- **37,044** articles from 2020
- **7** news outlets in 2020, ranging from Breitbart to NPR
- For each article we had
 - Title
 - Text
 - News outlet
 - Year

Approach Overview

Clustering

K-means
clustering

Agglomerative
clustering

BertTopic
Clustering

Sentiment Analysis

BERT

Vader

Analysis and Evaluation

Pre-Processing

Clustering

- For our clustering of news articles, we wanted the articles to maintain their natural structure and context, so we *did not* remove punctuation, stop words, etc. as might be common in other NLP tasks.
- Instead, we only removed certain non-text character codes (like unicode characters for newline, tab, etc.), and lowercased text.
- We also prepared representative text for each article, consisting of the concatenated article title and text, up to 512 characters.

Sentiment Analysis

- For the sentiment analysis, we did not change the native uppercasing/lowercasing, fixed some systematic raw-data errors (i.e, missing spaces between sentences), removed articles without text, and removed quotes.



Modeling



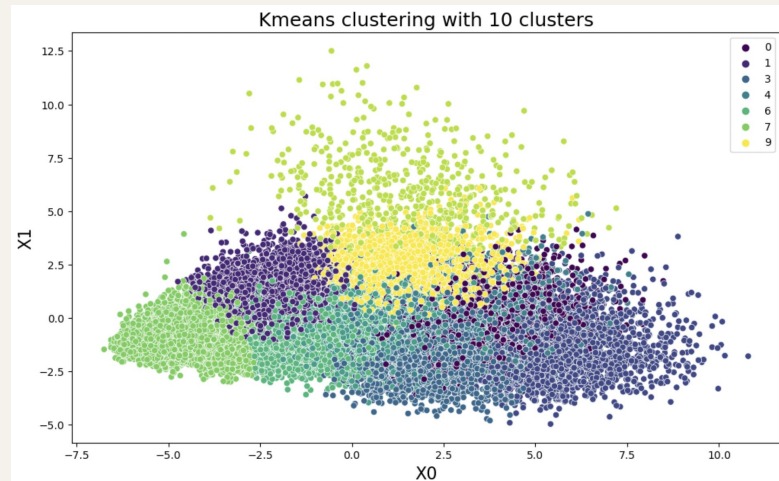
Modeling

Topic Clustering

K-Means

Partitions data into k number of clusters:

- 1) Basic idea of clustering: group similar objects into groups.
- 2) K is a hyperparameter
- 3) K-means clustering will try to minimize the sum of the squared distance between the vector representation of one object and the centroid of the cluster.



K-Means

High level idea of k-means algorithm:

- 1) Start with initial set of k-means $\mu_1^{(1)}, \mu_2^{(1)}, \dots, \mu_k^{(1)}$
- 2) Assign each vector to closer cluster mean
- 3) Recompute each cluster centroid $\mu_1^{(i)}, \mu_2^{(i)}, \dots, \mu_k^{(i)}$
- 4) Repeat until reaching convergence

Initialization is a highly important component of k-means clustering

- Initialization will impact the algorithm running time and quality of clusters
- Initialized using k++

K-Means

Embeddings:

- 1) Bert embeddings (Base, Uncased)
 - Transformer based
 - Highly contextualized
- 2) Term Frequency-Inverse Document Frequency(TF-IDF)
 - Measures how important a term is within a document with respect to the whole corpus of document

Run k-means clustering:

- k -values : 10, 20,50,100,250,500

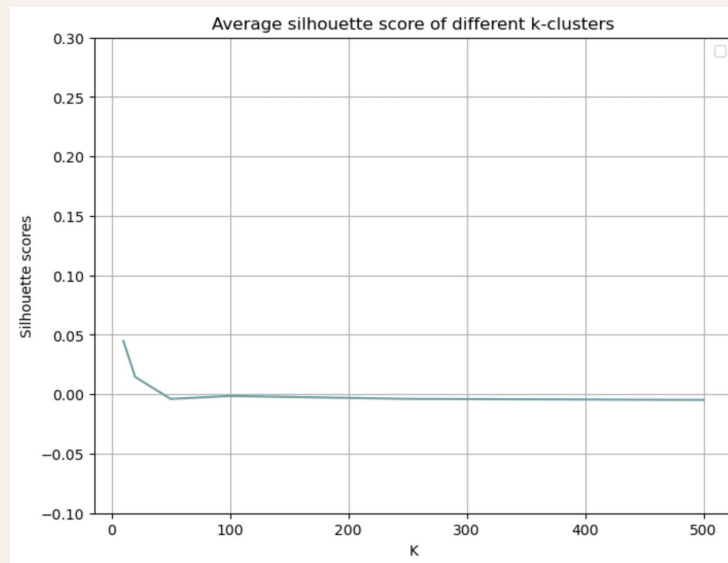
Retrieve most important features from each cluster using TF-IDF

K-Means

Given the lack of a true label for each cluster, we can't use usual methods for testing accuracy and instead relied on some unsupervised learning clustering metrics, particularly silhouette scores:

Silhouette scores: measurement to test separation of clusters by estimating how similar an object is to its own cluster compared to other clusters

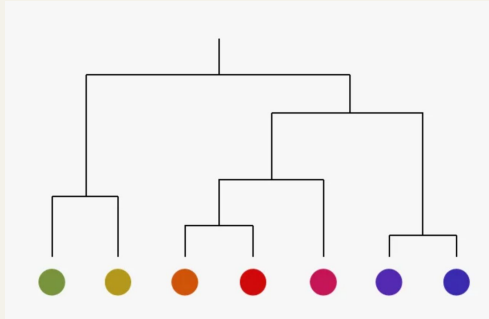
- Range from -1 to 1: Closer to 1 means we obtained strong clusters (good!), closer to 0 means significant overlapping between clusters (not good), closer to -1 means misclassifying (bad!)
- Results are indicative of overlapping clusters



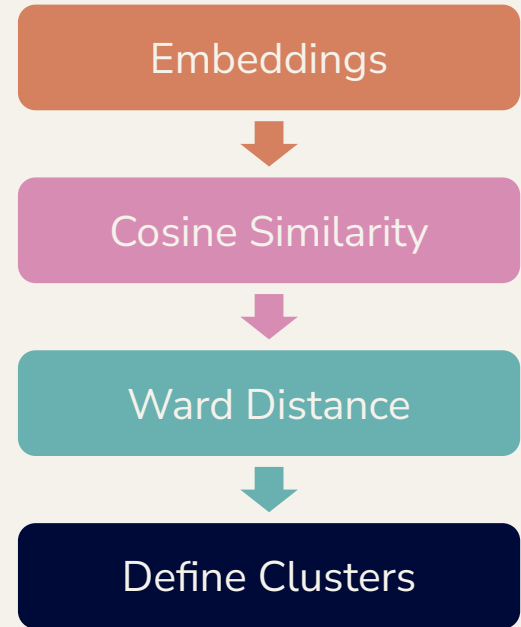
Hierarchical Clustering

Agglomerative clustering: start with many small clusters and merge them together to create many bigger clusters.

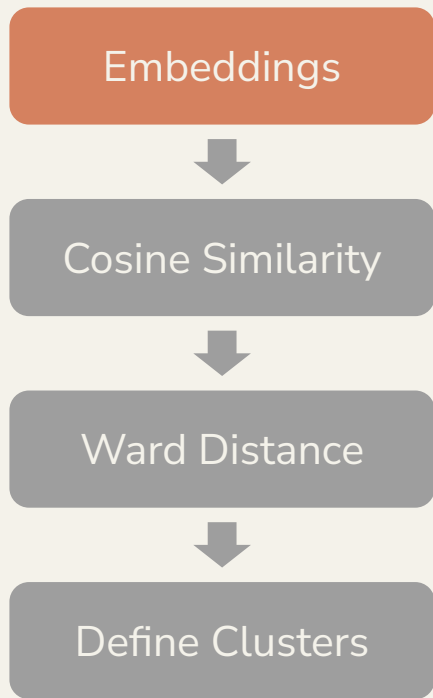
- 1) Define distances between all observations
- 2) Determine threshold to define clusters



Clustering Process



Hierarchical Clustering



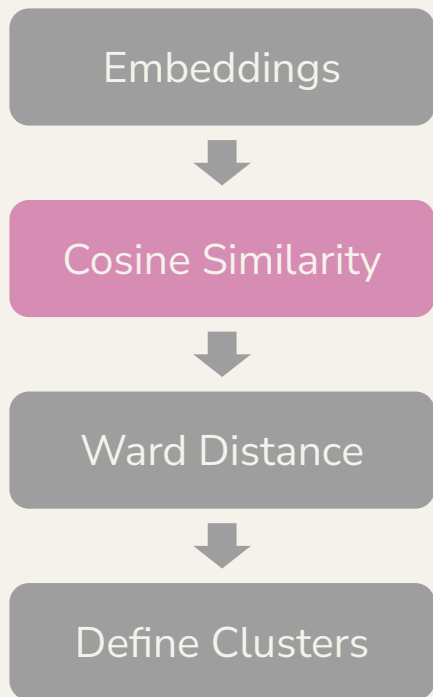
FastText

- Created by Facebook
- Uses the char-ngram components of words in combination with a word2vec word-level representation to detect morphological similarities
- Better than word2vec for handling out-of-vocabulary words
- Non-contextualized

BERT (Base, Uncased)

- Created by Google
- Transformer based model
- Highly contextualized

Hierarchical Clustering



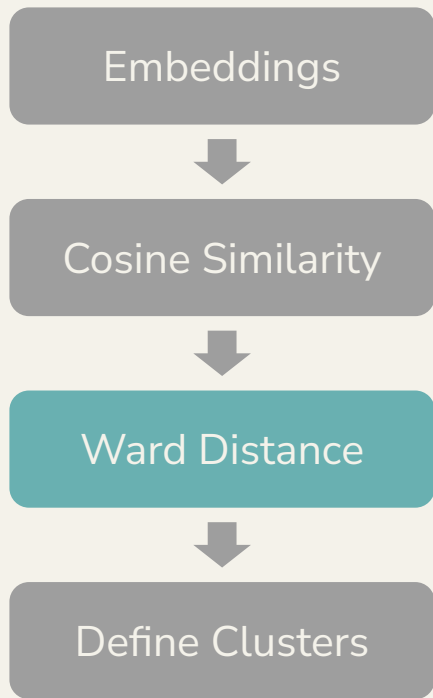
$$\cos(\theta) = \frac{A \cdot B}{||A|| ||B||} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Computes the cosine of the angle between two embedding vectors

i.e. two overlapping vectors will have the highest value and two opposite vectors will have the lowest value

* Because it measures similarity in terms of angles instead of raw distance this is compatible with PCA and dimensionality reduction.

Hierarchical Clustering



- Ward's minimum variance method: minimizes within-cluster variance
- Recursively merges the pair of clusters that minimally increases the within-cluster difference
- Useful for topic clustering because the hope is that similar topics will be near each other and won't be forced into a predefined number of clusters.

Hierarchical Clustering

Embeddings



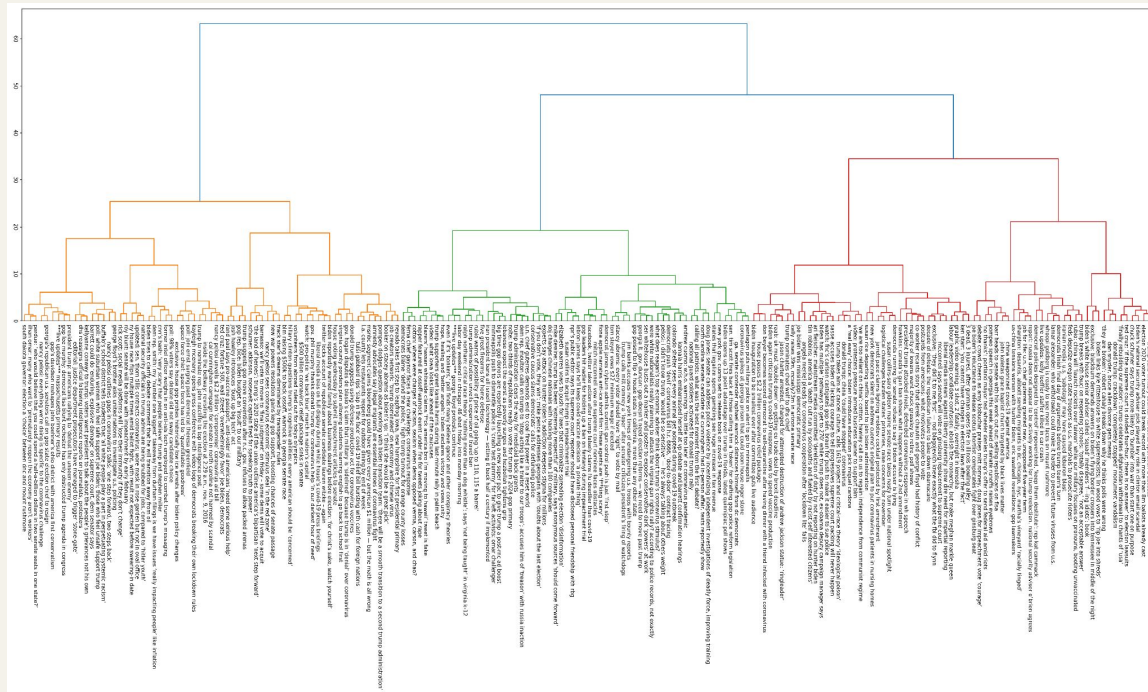
Cosine Similarity



Ward Distance

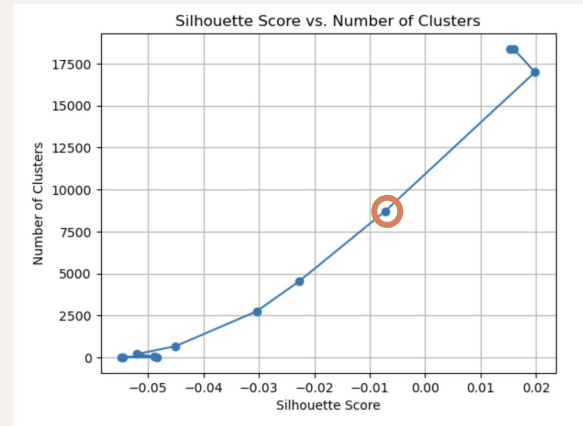
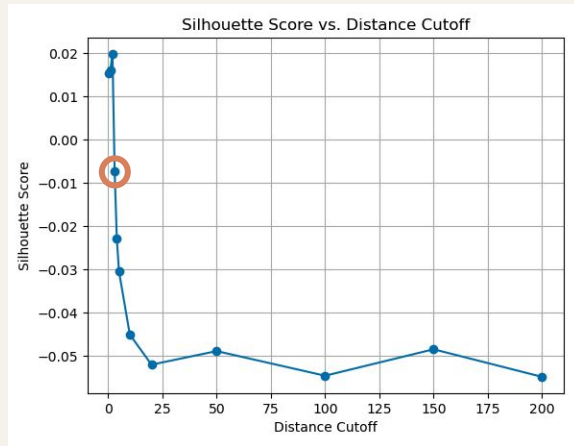
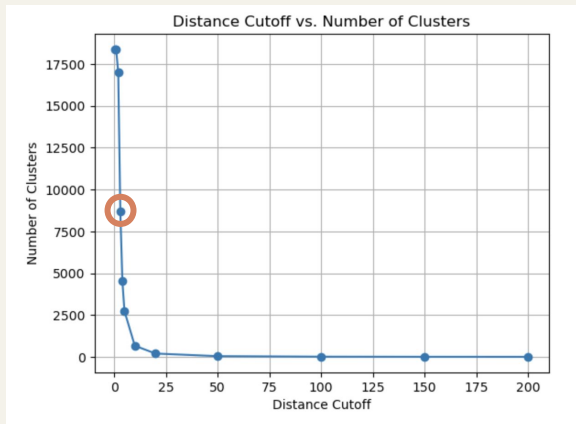


Define Clusters



FastText dendrogram

Hierarchical Clustering Results

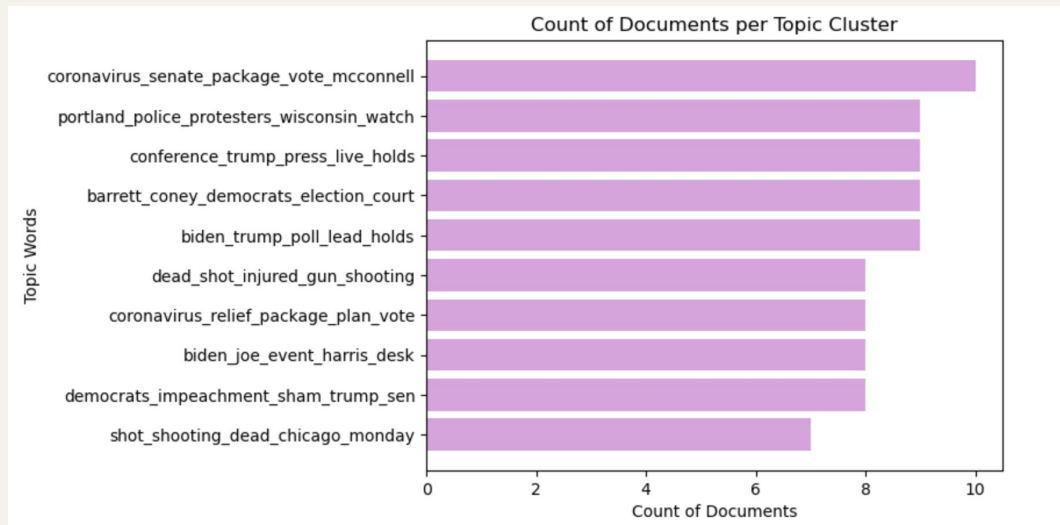


Hierarchical Clustering Results

Cluster cutoff: 3

Number of clusters: **8,711**

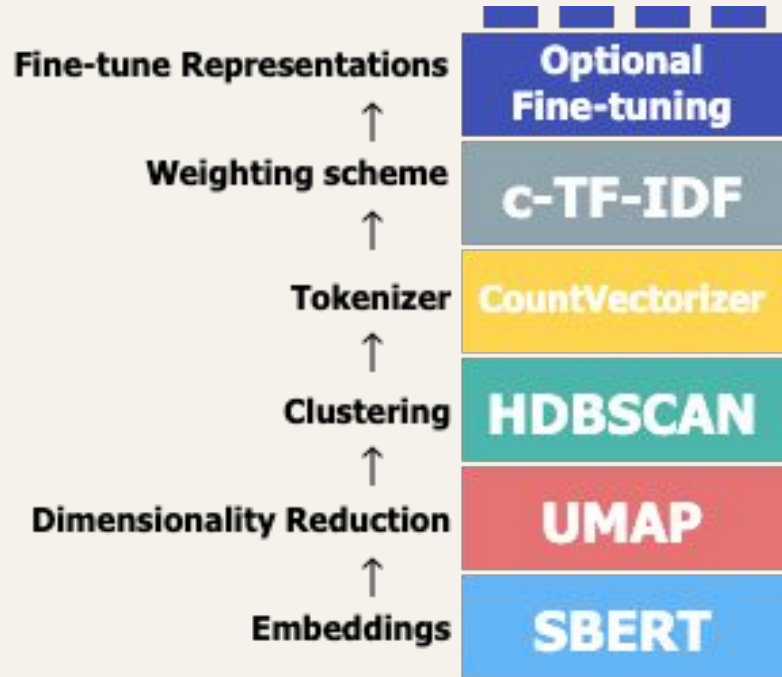
Silhouette Score: **-0.0072**



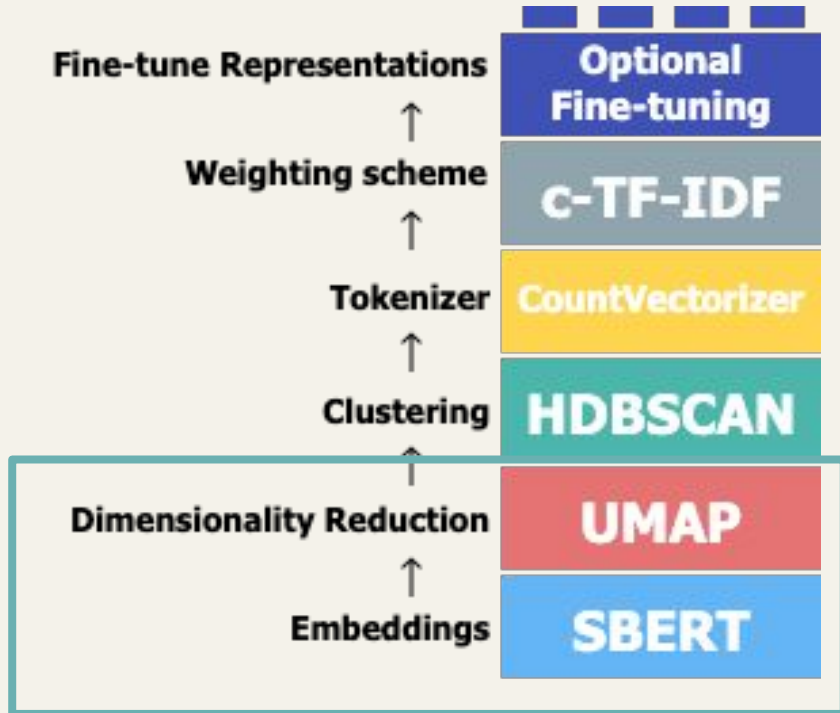
Labels created using TF-IDF per-cluster

BERTTopic

BERTTopic is a modular software package performs topic modeling on unlabeled text data. In our implementation, we use BERT sentence transformer to make embeddings, and use hierarchical clusters and c-TF-IDF to identify article topics.

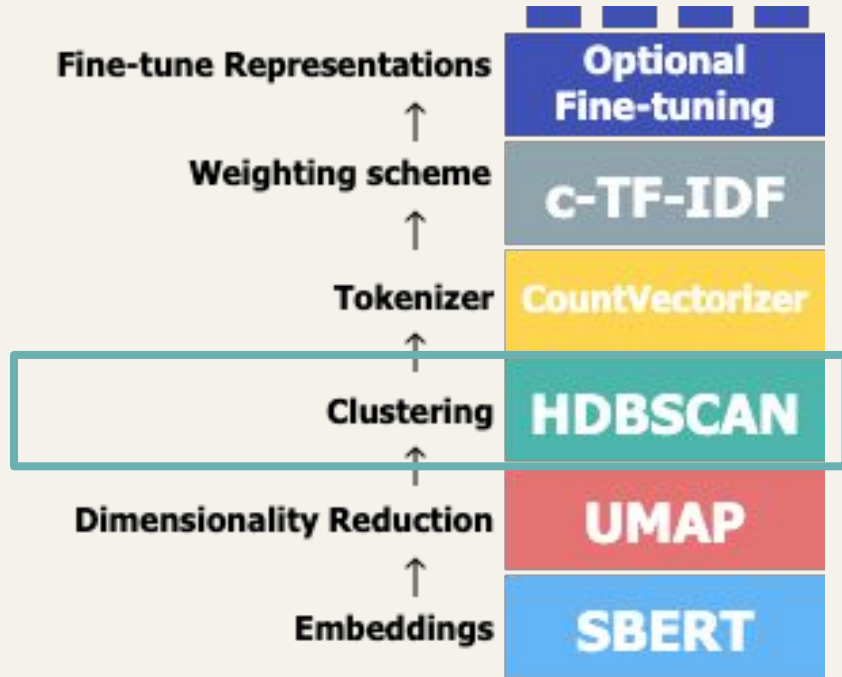


BERTTopic



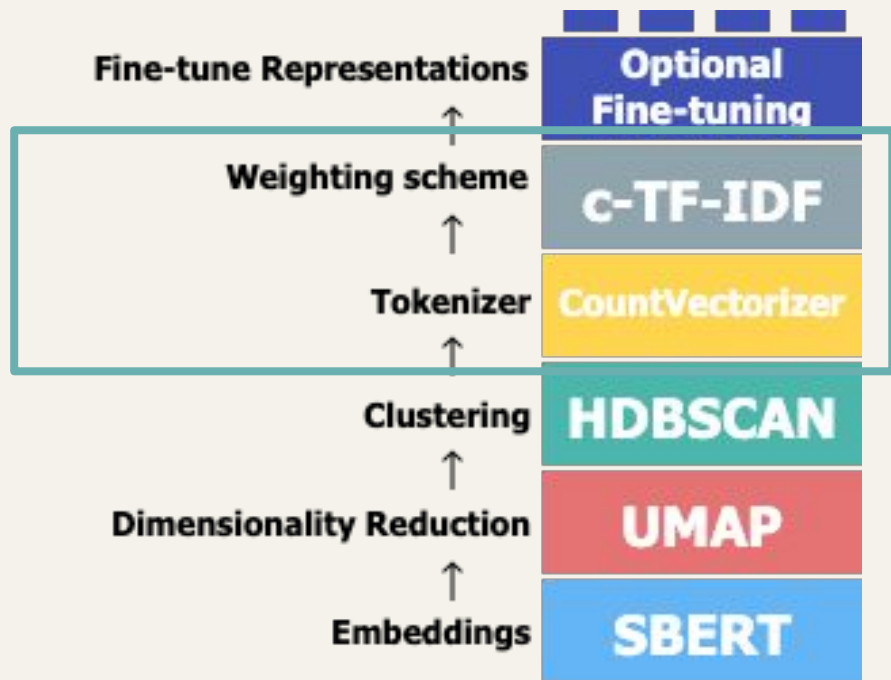
- To transform raw articles into dense vector representations, we use sentence-transformers ("all-MiniLM-L6-v2") which generally works well at capturing semantic similarity across sentences, while being 5x faster than the full base model.
- To reduce the dimensionality of the embeddings to make them compatible with clustering, as a default BERTTopic uses UMAP because it captures both the local and global high-dimensional space in lower dimensions.

BERTTopic



- By default, HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) is used to find clusters because it works well in finding clusters of different shapes and densities.
- “HDBSCAN works by first building a condensed tree, which is a hierarchical representation of the data points. The condensed tree is built by repeatedly merging data points that are close together. The algorithm then uses the condensed tree to identify clusters of data points. Clusters are identified by finding groups of data points that are close together in the condensed tree.”

BERTTopic



- To represent the clusters as topics, we first make a bag-of-words representation of all articles in the cluster.
- To define clusters in terms of what makes them unique from other clusters, the algorithm performs TF-IDF across clusters.
- The most important words per cluster are extracted to get descriptions of topics.

c-TF-IDF

For a term x within class c :

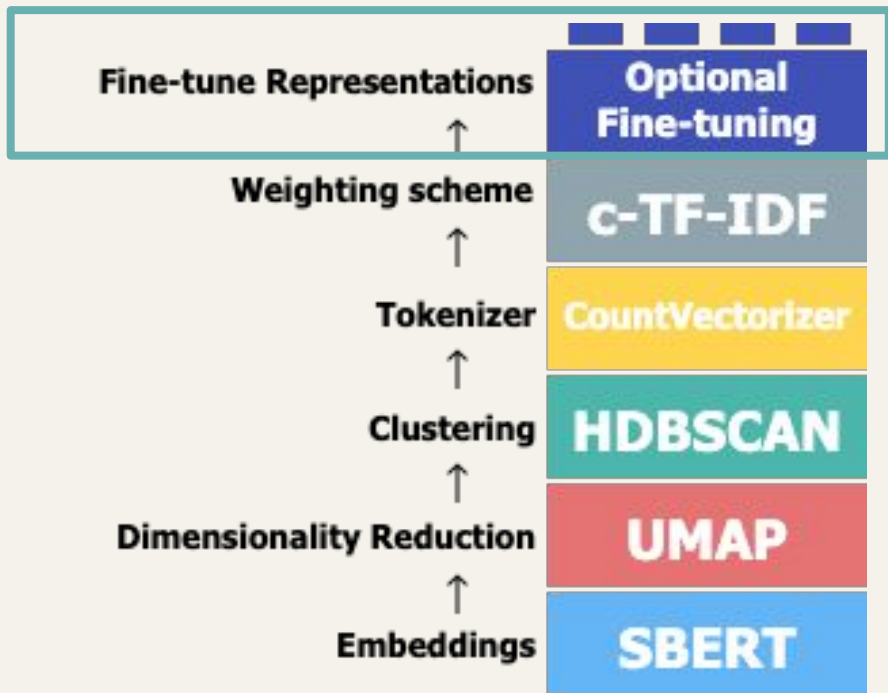
$$W_{x,c} = \| \text{tf}_{x,c} \| \times \log \left(1 + \frac{A}{f_x} \right)$$

$\text{tf}_{x,c}$ = frequency of word x in class c

f_x = frequency of word x across all classes

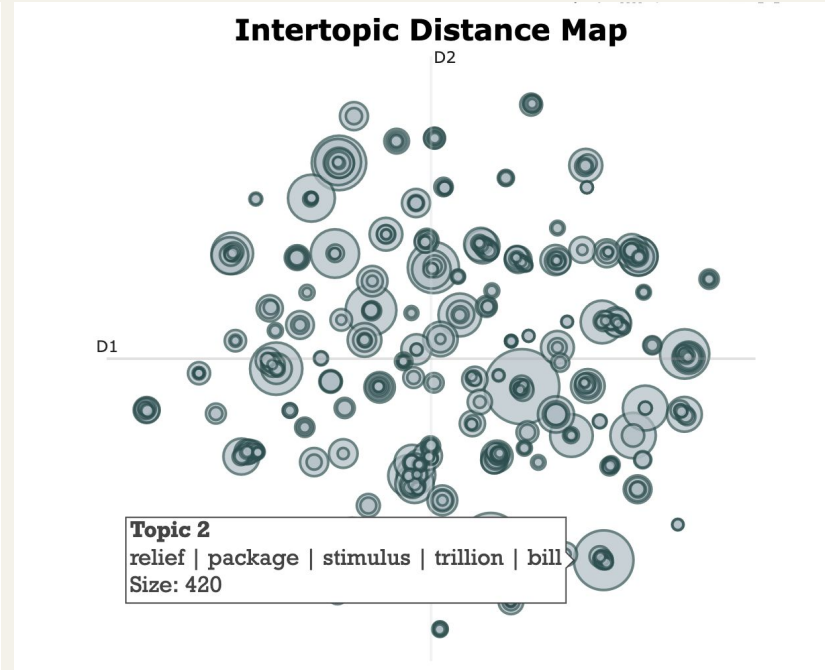
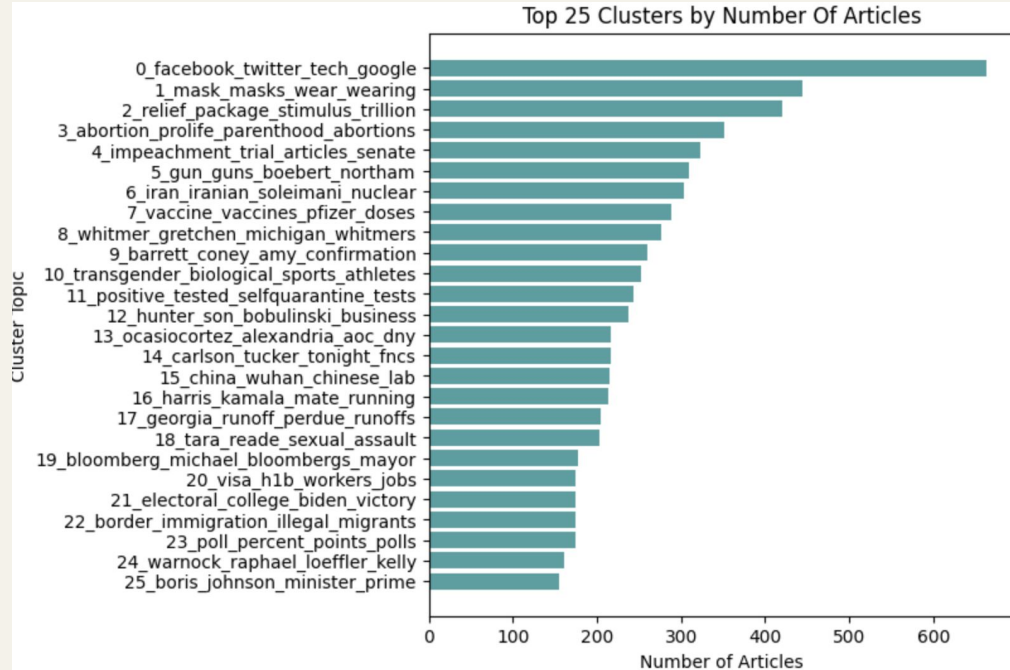
A = average number of words per class

BERTTopic



- We experimented with fine tuning representations by implementing KeyBERTInspired, which aims to improve coherence and remove stopwords from the topic representations.
- KeyBERTInspired considers the semantic relationship between keywords within a given cluster and calculates similarity between candidate keywords and the topic embeddings.
- Given the interpretability of the starting clusters, we did not implement KeyBERTInspired in the final model.


BERTTopic: Results





Cluster Performance Discussion

- BERTTopics created the most coherent, interpretable clusters. However, in not limiting the cluster numbers, we ended up with a very large number of clusters, some of which had very few articles- this made sentiment analysis on the clusters challenging.
- Relative to K-means, we hypothesize that BERTTopic may have performed better because, in K-means, you may force every article to be in a cluster; when you force all articles to be in a cluster, the clusters will be noisy, which can make topic representations incoherent.



Modeling

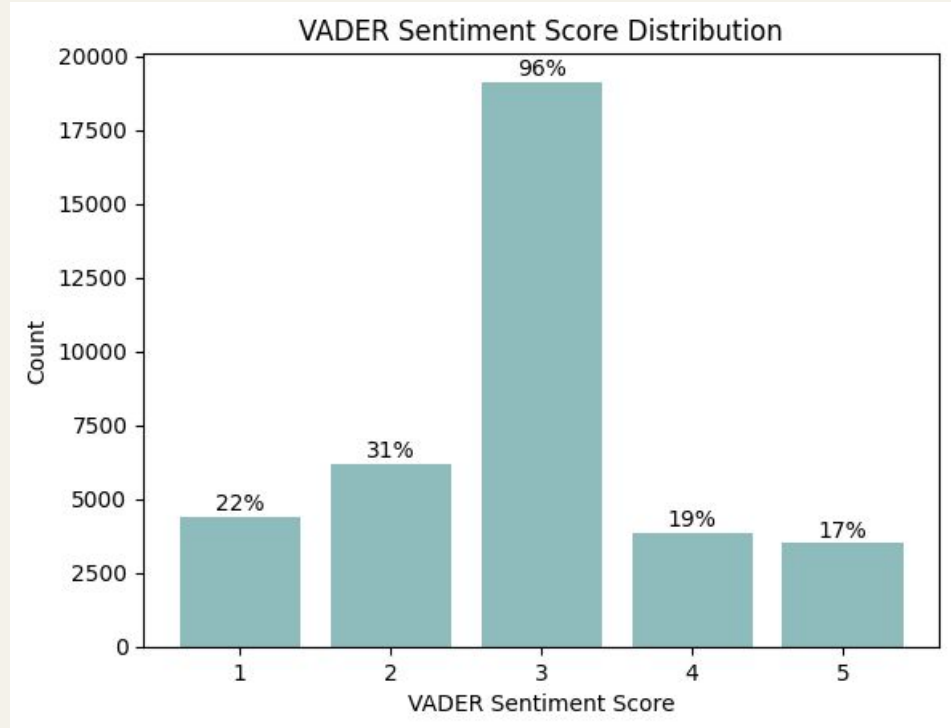
Sentiment Analysis

VADER

- VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is classifies sentiment between -1 and 1.
- Not a machine learning model – instead it's a dictionary which maps words and other lexical features to sentiment scores. The dictionary was developed by asking contractors from the Amazon Mechanical Turk to rate the sentiment of a word and averaging the scores.
- VADER also considers these heuristics:
 - Punctuation (. vs !)
 - Capitalization (great vs GREAT)
 - Degree modifiers (very vs sort of)
 - Shift in polarity due to but (I love you, but...)
- Pro's:
 - Very fast
 - Trained by humans
- Con's:
 - Not all words are scored
 - Limited contextual understanding

VADER Classification

- Vader model returns a probability that the statement is positive, neutral, or negative and weighted compound score.
- Use compound score as continuous variable in analysis
- Typically classification thresholds at -0.5 and 0.5
- Added additional thresholds at -0.3 and 0.3 to create five point scale with more variation in polarity



VADER Examples

Text	VADER Classification
'off the rails: moderators struggle as 10th democratic debate turns into absolute shouting match'	Compound Score: -0.3182 Negative
"Trump Didn't See It Coming: Coronavirus Deaths Increased Tenfold This Month"	Compound Score: 0.2732 Somewhat Positive
'Cory Booker: Clearly Amy Coney Barrett Is Not a Racist'	Compound Score: 0.7114 Positive
Delingpole: Dawn Butler MP Is Not a Victim of Police Racism	Compound Score: -0.5083 Negative

BERT For Sentiment Analysis

- BERT can also be used for sentiment analysis
- Pros:
 - Better ability to understand context and to handle negation and ambiguity.
 - BERT can handle new vocabulary better than VADER
- Cons:
 - Won't be able to finetune model since our data is unlabeled
 - BERT models trained on similar data typically predict positive or negative sentiment which would not allow us to analyze the variance in the sentiments

Fine tuning BERT with VADER labels?

- Used VADER to label the sentiment and then fine tuned a pre-trained BERT model
- Saw this method suggested on a few blogs
- This should help address the issues of VADER's limited vocabulary and contextual understanding.

BERT Model

- Used pretrained 'distilbert-base-uncased' tokenizer and 'Distill Bert For Sequence Classification' model
- Modified model to predict one of 5 labels and use cross-entropy loss which is better suited to multi-class classification mean-square loss.
- Random sample of 20,000 titles (16,000 in training set and 4,000 in test set)
- Pre-trained Model Configuration:
 - Number of attention heads: 12
 - Number of transformer layers: 6
 - Activation Function: non-linear function, GELU (Gaussian Error Linear Unit)
 - Masking: Yes
 - Dropout: Yes
- Trained model using the following hyper-parameters:
 - Epochs: 3
 - Batch size: 16
 - Learning Rate: $5e-5$ with 30 warm-up steps
 - Weight Decay: 0.01
 - Evaluates after 500 steps

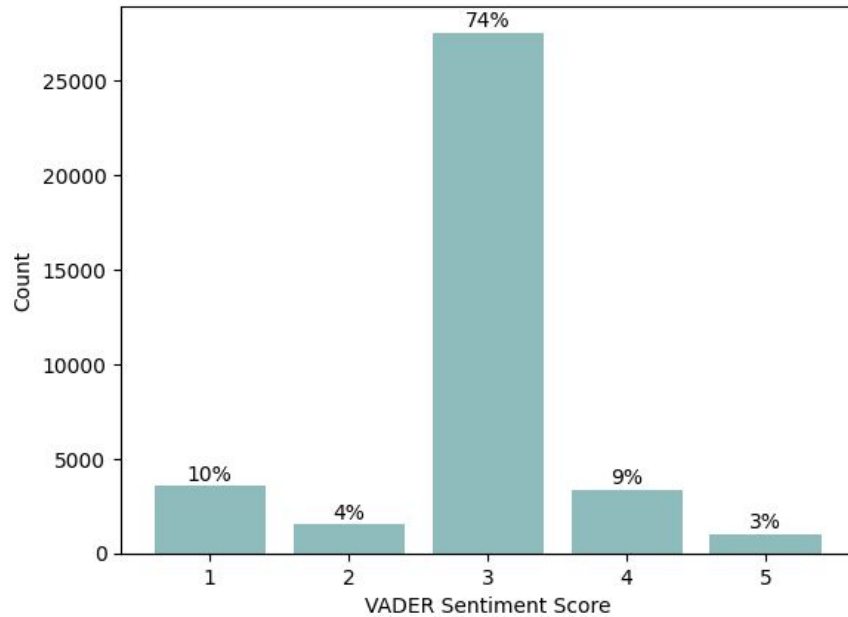
Is the BERT model better?

- Evaluation Accuracy = 0.82875 and Evaluation F1 = 0.8261998804110827, but is this good if the goal is to improve on the VADER model?
- Changed the label on **3,798** articles
- Difficult to evaluate since our data is unlabelled. Medium post on a test of this method found BERT was 2% more accurate than VADER alone in predicting the sentiment of reviews.

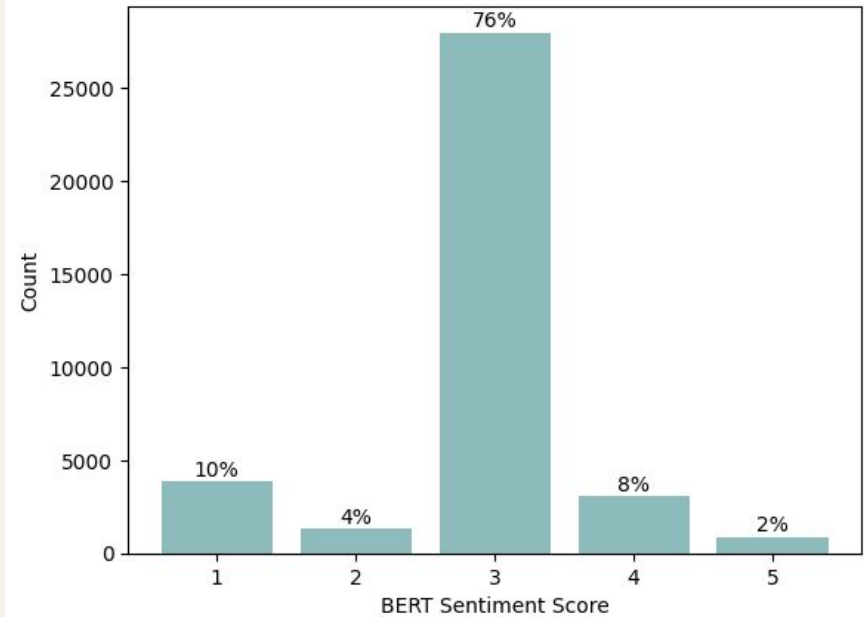
Text	VADER Classification	BERT Classification
"Trump Didn't See It Coming: Coronavirus Deaths Increased Tenfold This Month"	Somewhat Positive Compound Score: 0.2732	Neutral (change of -1)
'Donald Trump: Facts Are on Our Side in Election Fight, but Time Isn't'	Neutral Compound Score: -0.2023	Negative (change of -2)
Delingpole: Dawn Butler MP Is Not a Victim of Police Racism	Negative -0.5083	Positive (change of +4)

Model Comparison

VADER Sentiment Score Distribution



BERT Sentiment Score Distribution





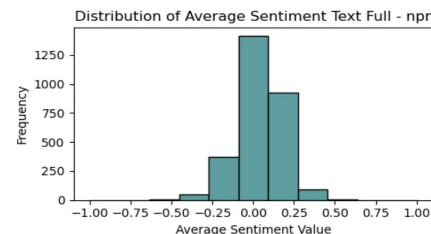
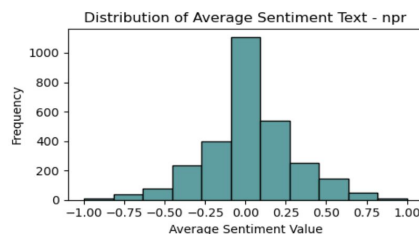
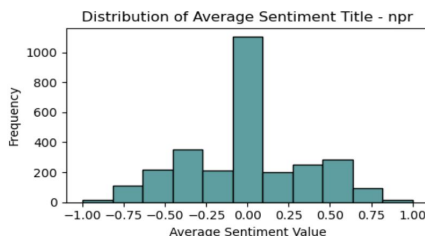
Results

Sentiment Analysis of Clusters

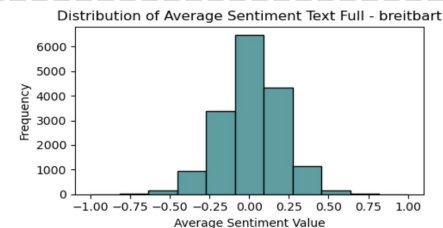
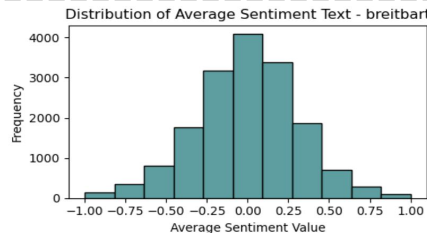
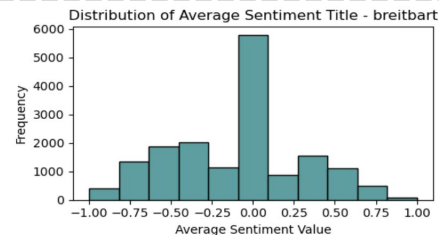
VADER Classification of News Sources

- We did an initial analysis of articles using VADER before creating clusters
- Key findings:
 - titles tend to be neutral
 - full articles have less variation than the title and first part of articles text
 - News sources have modest variations in their overall distributions
- Rest of the analysis will be on the title + article text up to 512 words

NPR VADER
Score
Distributions:

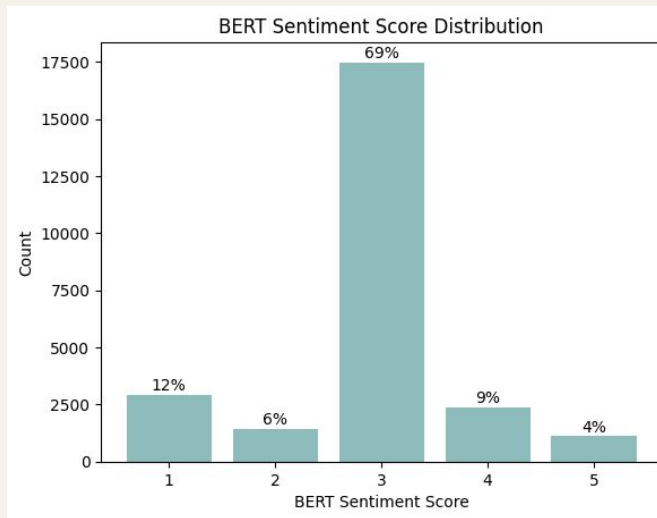
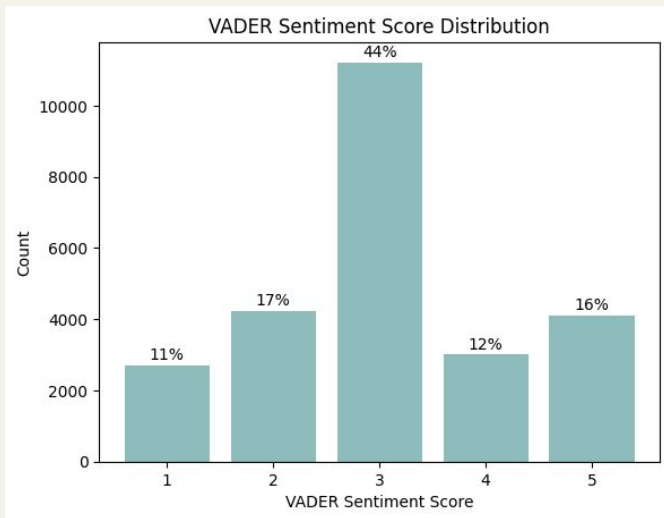


Breitbart VADER
Score
Distributions:



BERT Topic Clusters

- Dropped the outlier articles which left **25308 articles** and **544 topic clusters**
- Many of the outlier articles were classified as neutral which shifted the distribution of our sentiment scores to be less heavily neutral
- Average VADER sentiment per cluster: **-0.0045**
- Minimum VADER sentiment per cluster: **-0.73**
- Maximum VADER sentiment per cluster: **0.77**



Top Topics by Sources

NPR

1. **voting_cast_registration_voter**
2. facebook_twitter_tech_google
3. **census_hansi_bureau_wang**
4. mask_masks_wear_wearing
5. relief_package_stimulus_trillion

Fox

1. facebook_twitter_tech_google
2. mask_masks_wear_wearing
3. relief_package_stimulus_trillion
4. vaccine_vaccines_pfizer_doses
5. barrett_coney_amy_confirmation

Breitbart

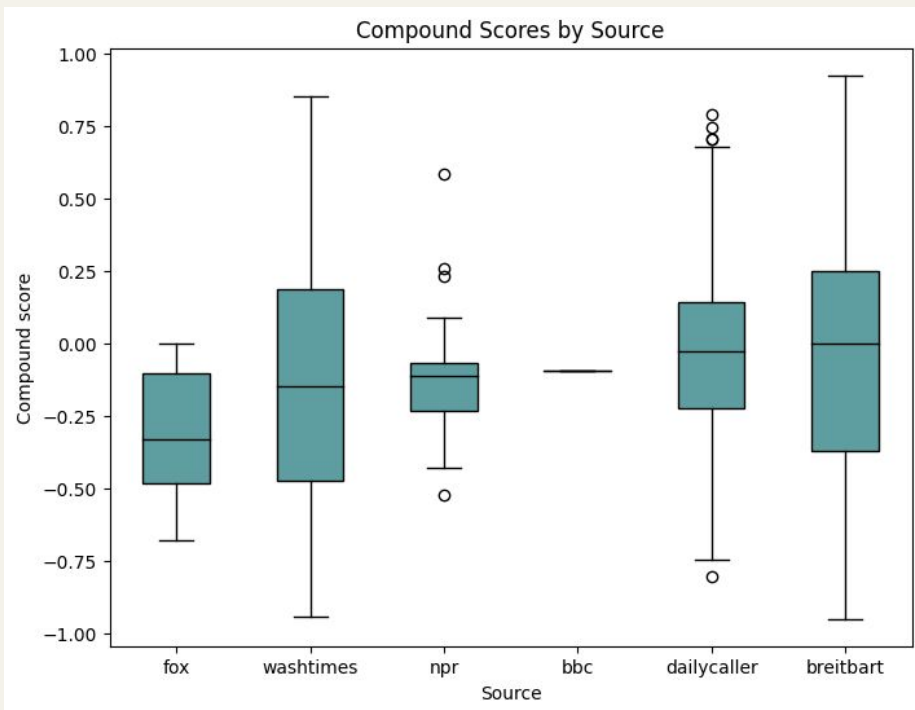
1. facebook_twitter_tech_google
2. mask_masks_wear_wearing
3. **gun_guns_boebert_northam**
4. **iran_iranian_soleimani_nuclear**
5. **transgender_biological_sports_athletes**

Daily Caller

1. facebook_twitter_tech_google
2. relief_package_stimulus_trillion
3. **abortion_prolife_parenthood_abortions**
4. **positive_tested_selfquarantine_tests**
5. **tara_reade_sexual_assault**

*unique topics are in bold

Impeachment



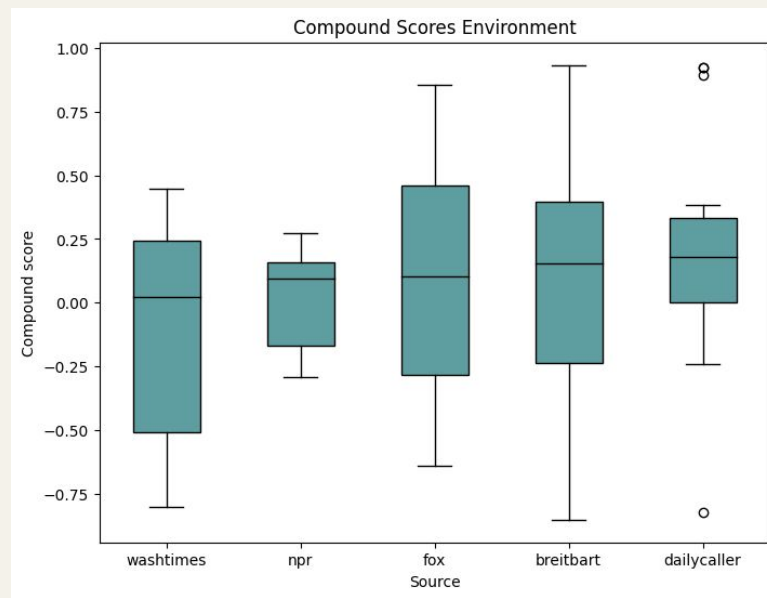
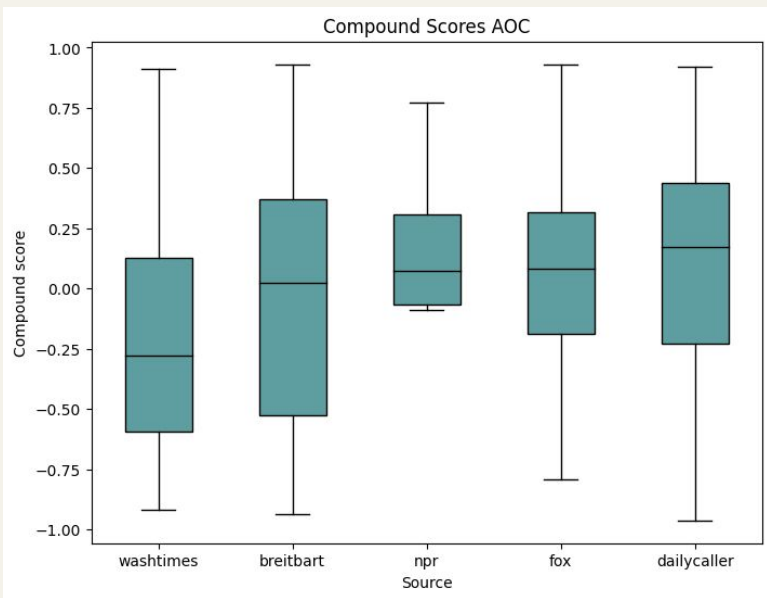
Breitbart – VADER: -0.9493, BERT: 2

'charles hurt: after impeachment farce ends, next comes 14th amendment with the result of the second impeachment charade a forgone conclusion in washington, let us now turn to the final partisan tool remaining in democrats' torture closet: the 14th amendment.this amendment allows congress to banish from future federal office any current or former politician deemed by congress to have "engaged in insurrection." its adoption after the civil war was designed to prevent officers or political leaders of the..'

Breitbart – VADER: 0.8790, BERT: 4


'jim jordan: senate impeachment trial to end soon, american people 'don't seem to be tuning in' during a friday interview with fox news channel's "america's newsroom," rep. jim jordan (r-oh) sounded off on the senate impeachment trial.jordan predicted the impeachment trial, which he noted nobody is tuning into, will end in the next week so congress can get back to working for the american people. "[t]hese facts are so strong for the president, i feel real confident that hopefully next week we'll get this..'

Topics





Future Work

- **K-means:** explore kernel functions to capture higher dimensional separability
 - **Hierarchical:** explore non-flat clustering schemes
 - **BertTopic:** fine-tuning for fewer clusters
 - **Sentiment:**
 - Experiment with other pre-trained BERT models
 - Label data
 - **Data**
 - Rerun the model on a larger sample
 - Compare topics between years
 - More comprehensive news source coverage
- 



Questions?



Thank You!

Bert Training - Titles

Step	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
500	0.965300	0.694121	0.738000	0.734882	0.733638	0.738000
1000	0.660300	0.558255	0.794000	0.784106	0.788634	0.794000
1500	0.431700	0.523832	0.817000	0.817472	0.822690	0.817000
2000	0.393900	0.435232	0.842500	0.843947	0.846790	0.842500
2500	0.224000	0.513907	0.837250	0.836957	0.838295	0.837250
3000	0.211300	0.489216	0.851250	0.852294	0.854224	0.851250