Online Appendix

Selecting More Informative Training Sets with Fewer Observations

Aaron R. Kaufman[*]

Contents

[*]Division of Social Science, New York University Abu Dhabi, aaronkaufman@nyu.edu

1 Representations and Subset Selection Methods

Here we define each of the representations and subset selection methods we test.

1.1 Low-dimensional Representation

(i) **Multinomial Topic Factor-based weight vectors (from the baseline):** A K-topic model represents each document (given as vector of token counts) in the dataset as a sample from multinomial distribution

$$x_i \sim \mathrm{MN}(w_{i1}\theta_1 + \ldots + w_{iK}\theta_K, m_i),$$

where $\theta_j \in \mathbb{R}^p$ is the j-th topic vector (same for all the documents), p is the number of tokens in the vocabulary, $w_{ij} \in \mathbb{R}$ is the weight of j-th topic in i-th document, m_i is the length of x_i

Parameters θ_j and w_{ij} are estimated using MAP with Dirichlet priors:

$$w_{ij} \sim \mathrm{Dir}(1/K)$$

$$\theta_j \sim \mathrm{Dir}(1/(Kp))$$

Vector of topic weights $w_i = (w_{i1}, \ldots, w_{iK})$ is used as low-dimensional representation of the document x_i (Taddy 2012).

(ii) **Sentence-level RoBERTa:** RoBERTa-base model was used to encode the documents into dense vectors of size 768 each. The model is trained on NLI and STSb datasets, uses mean pooling over words and encodes 830 sentences per second on V100 GPU (Reimers and Gurevych 2019).

(iii) **Sentence-level DistilBERT:** DistilBERT-base model was used to encode the documents into dense vectors of size 768 each. The model is trained on NLI and STSb datasets, uses mean pooling over words and encodes 4000 sentences per second on V100 GPU (Reimers and Gurevych 2019).

(iv) **Sentence-level GloVe6B:** GloVe6B model was used to encode the documents into dense vectors of size 300 each. It is trained on Wikipedia 2014 and Gigaword 5 (6 billion tokens, 400k vocabulary), uses mean pooling over words and encodes 34000 sentences per second on V100 GPU (Reimers and Gurevych 2019).

(v) **Universal Sentence Encoder:** It encodes sentences into embedding vectors that specifically target transfer learning to other NLP tasks, thus, the embeddings generated can be used for multiple-tasks. There are two encoder models - transformer with self-attention and deep averaging network that computes unigram and bigram embeddings first and then averages them to get a single embeddings. The encoded vectors have 512 dimensions (Cer et al. 2018).

(vi) **t-SNE dimension reduction on BoW:** t-Distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction algorithm. First it constructs a distribution over pairs of objects in the initial high-dimensional space in such a way that a pair of close objects has higher probability than a pair of less similar objects. Then, it constructs a similar distribution in the low-dimensional space and minimizes the KL divergence between these two distributions. The objects that were close in the initial space remain close in the lower-dimensional space (Maaten and Hinton 2008).

(vii) **PCA dimension reduction on BoW:** PCA or principal component analysis identifies patterns in data based on the correlation between features. PCA aims to find the directions of maximum variance in high-dimensional data (along orthogonal axes called principal components) and projects it onto a new subspace with equal or fewer dimensions than the original one. This new subspace includes the top-k principal components if the dimensions after reduction are chosen to be k.

(viii) **UMAP dimension reduction on BoW:** Uniform Manifold Approximation and Projection (UMAP) is a dimensionality reduction technique that can be used for general non-linear dimension reduction. The manifold is modeled with a fuzzy topological structure. The reduced embedding is found by searching for a low dimensional projection of the data that has the closest possible equivalent fuzzy topological structure.

(ix) **NMF dimension reduction on BoW:** The logic for dimensionality reduction with Non-negative matrix factorization (NMF) is to take the $m \times n$ data and to decompose it into two matrices of dimensions $m \times features$ and $features \times n$ respectively. The $features$ will be the reduced dimensions.

Note: all of our Bag-of-Words representations considers unigrams, bigrams, and trigrams, and as pre-processing, removes stopwords and punctuation, and removes words occurring in fewer than 10 of 10,000 documents.

1.2 Representative Subset Selection

(i) **D-Optimal Design Subset Selection (from baseline):** D-Optimal design is an iterative greedy algorithm that on each step selects one more observation that maximizes the increase of information determinant $D_t = |X_t^T X_t|$, where $X_t = (x_{i1}, \ldots, x_{it})^T$ is the matrix of selected observations (Taddy 2013).

Information determinant on the step $t+1$ can be written as

$$D_{t+1} = |X_t^T X_t + x_{t+1}^T x_{t+1}| = D_t \left(1 + x_{t+1^T} \left(X_t^T X_t\right)^{-1} x_{t+1}\right)$$

The observation on step $t+1$ is selected as

$$x_{t+1} = \arg\max_{x \notin X_t} x^T \left(X_t^T X_t\right)^{-1} x$$

(ii) **Random Pick:** As the name suggests, in this method the required number of documents are selected at random (Pan et al. 2005).

(iii) **K-means Clustering-based selection:** Given a set of documents encoded into vectors $(x_1, x_2, \ldots, x_n)$ where each document is a d-dimensional vector, the k-means clustering algorithm aims to partition the n documents into k clusters $C_1, C_2, \ldots, C_k$ by minimizing the intra-cluster distances (Daszykowski, Walczak, and Massart 2002). The algorithm starts at $t = 1$ with k centroids (means) $(m_1^{(t)}, m_2^{(t)}, \ldots, m_k^{(t)})$ and proceeds by alternating between two steps:

Assignment step: Each document is assigned to the cluster with the nearest centroid (with Euclidean distance as the distance measure).

$$C_i^{(t)} = \{x_p : ||x_p - m_i^{(t)}||^2 \leq ||x_p - m_j^{(t)}||^2 \; \forall j, 1 \leq j \leq k\}$$

Update step: The centroids are recalculated based on the documents in the cluster.

$$m_i^{(t+1)} = \frac{1}{\left|C_i^{(t)}\right|} \sum_{x_j \in C_i^{(t)}} x_j$$

The algorithm converges when the assignments no longer change. Next, from each cluster, one document is selected in order to get a set of k distant documents.

(iv) **Greedy Farthest Point Sampling using Kullback-Liebler Divergence or KL Divergence:** KL Divergence or relative entropy is a measure of how one probability distribution is different from another. The idea is to identify the documents for which the probability distributions across features are maximally different from each other. This method is inspired from the relative entropy-based subset selection method (Pan et al. 2005). In our case, the vectors are dense, continuous, and real-valued. Hence, the algorithm from the paper is not applicable as is and has very long execution time. Our algorithm proceeds as follows:

- Fit a known distribution to each observation. In our case, since most of the representations have normal distribution, we fit the same to each observation using `scipy` module's `norm.fit` method and obtain corresponding mean and standard deviation.

- Calculate KL Divergence between each pair of observations using:

$$KL(p, q) = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

  KL divergence is asymmetric, but from observation, $KL(p, q)$ is close to $KL(q, p)$. In order to use KL-divergence as a distance measure (and for creating distance matrix), we sum $KL(p, q)$ and $KL(q, p)$.

- Use greedy farthest point sampling, as described in Algorithm 1, using the distance matrix obtained in the previous step as the input.

```
Input: Distance matrix 'D[0...n-1, 0...n-1]' of size nxn, number of observations k;
Result: Array 'indices[0...k-1]' of size k
Initialization: indices[0...k-1] = 0;
//select two points at maximum distance;
indices[0], indices[1] = $\arg\max_{i,j} D[i,j]$;
while $i¡k$ do
    // select the point which maximizes the minimum distance from already selected points;
    indices[i] = $\arg\max_{j \notin indices}(\min_{idx \in indices}(D[idx, j]))$;
    i = i + 1
end
```

Algorithm 1: Greedy Farthest Point Sampling

(v) **Greedy Farthest Point Sampling using two-sample Kolmogorov-Smirnov statistic:** The two-sample Kolmogorov–Smirnov statistic (KS-statistic) quantifies a distance between the empirical distribution functions of two samples. The null distribution of this statistic is calculated under the null hypothesis that the samples are drawn from the same distribution (in the two-sample case). The two input embeddings are sorted and converted to two cumulative diftribution functions $F_1(x)$ and $F_2(x)$ and the KS-statistic is calculated as:

$$KS(F_1, F_2) = \sup_x |F_1(x) - F(x)|$$

Using above, a pairwise distance matrix is obtained. Then, greedy farthest point sampling, as described in Algorithm 1 can be used to obtain the most distant points that are representative of the subset.

(vi) **Greedy Farthest Point Sampling using cosine distance:** Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space. It is defined to equal the cosine of the angle between them, which is also the same as the inner product of the same vectors normalized to both have length 1. Cosine distance is calculated by subtracting cosine similarity from 1. The cosine distance between two vectors or embeddings A and B is given by

$$D_{cos}(A, B) = 1 - \frac{\vec{A} \cdot \vec{B}}{||\vec{A}|| \times ||\vec{B}||}$$

Using above, a pairwise distance matrix is obtained. Then, greedy farthest point sampling, as described in Algorithm 1 can be used to obtain the most distant points that are representative of the subset.

(vii) **Reconstruction Loss Minimization with Sparsity Regularization:** This is based on a similar intuition as using an auto-encoder for feature selection (Han et al. 2018). Here, we attempt to reconstruct the entire dataset using a weight matrix while applying sparsity regularization on the weights in order to limit the amount of data that can be used for reconstruction of the original dataset (effectively disallowing learning of identity function). This allows us to identify the most relevant observations that can reconstruct the entire dataset. The steps are given below.

- Set X = transpose of complete dataset. It has d features and N observations The shape of Y is $d \times N$.
- Minimize reconstruction objective with sparsity regularization (Panda, Das, and Roy-Chowdhury 2016) (regularization term is the sum of L2 norms of rows of weight matrix W (size: $N \times N$) multiplied by λ- the regularization coefficient):

$$\phi(W) = \frac{1}{2}||Y - YW||_F^2 + \lambda||C||_{2,1}$$

- Select rows/observations with higher L2 norm ($||C_i||_2$).

2 Simulation Design

As discussed in the main text, our simulation design consists of four steps: representation, train/test splitting, document selection, and modeling. We discuss each step in detail below.

We define our problem of training set construction as consisting of two parts: representing documents in a low-dimensional space, and then selecting documents that optimally cover that low-dimensional space. As such, we consider all combinations of a number of possible representations and a number of possible space-filling metrics. Then, a given method involves projecting a corpus of documents into that low-dimensional space, then algorithmically selecting the documents that span it.

In our first application (Executive Orders), our task is identifying whether an Executive Order is ceremonial (y=0) or policy-relevant (y=1). In the second (StockTwits), our task is identifying whether a social media post is optimistic/bullish (y=1) or pessimistic/bearish (y=0).

2.1 Representation

To produce representations, we take the full data sets of 10,726 observations for the unilateral actions corpus and the 6,264 observations for the StockTwits corpus and project them into each of our representations. In our replication materials, we perform this step in "Data Preprocessing.ipynb" for the bag-of-words representations and in "Embeddings.ipynb" for the BERT-based representations.

2.2 Train/Test Splitting

To obtain uncertainty estimates around each method's accuracy we randomly produce 10 different train/test splits, where 80% of the documents are in the training set: for the first data set, 2,044 documents are held out in each split; for the second data set, 1,253 are held out in each split. In our replication materials, we perform this step in the first half of "Indices.ipynb" and "Reconstruction Loss.ipynb".

2.3 Document Selection

Next, for each data set, each train/test split, each method, and each of the ten values of k indicating the number of training samples, we produce a set of indices for which documents each method has identified as being optimal to label. All together there are more than 12,000 sets of indices. In our replication materials, we perform this step in “Indices.ipynb” and “Reconstruction Loss.ipynb”.

2.4 Modeling

For each of the more than 12,000 sets of indices, we train a Multinomial Naive Bayes classifier, a model that works best with with discrete features (in our case, count vectors, i.e., the Bag-of-Words model). Multinomial Naive Bayes has one hyperparameter to tune: the Laplace smoothing parameter. We tuned it using 5-fold cross-validation; the optimal parameter value we used to compute evaluation metrics on the test set. The list of values we tested for tuning the smoothing parameter is [1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 2, 4, 6, 8, 10, 12, 14, 16, 18]. In our replication materials, we perform this step in “MultinomialNB.ipynb”.

2.5 Accuracy Scoring

We evaluate accuracy using AUROC, the area under the receiver-operator characteristic. Despite our application entailing class imbalance, we prefer AUROC to metrics like F1 for two reasons. First, AUC is calculated using predicted probabilities and therefore does not require setting a probability threshold, while F1 is calculated using predicted classes, and thus does require setting a probability threshold. Since we conduct so many simulations, setting probability thresholds would introduce new comparability problems across our simulations as the thresholds would likely be different for different methods and across different iterations. Secondly, AUC is a more interpretable quantity of interest than F1, since it indicates the probability that a random observation in the positive class (y=1) will have a higher predicted probability than a random observation in the negative class (y=0).

3 Additional Results: Comparing the Different Methods

How different are these competing methods? We answer these methods in three ways. First, we compare their accuracy (as measured by AUC) as the number of training samples increases. Second, to interrogate these differences, we look for overlap in which observations each method selects for the training set when $k = 1000$, shown in a heatmap below in Figure 3. Finally, we compare how long each method takes to produce a training set.

Figure 2 in the main text shows AUC for the simple random sample approach, the Taddy (2013) approach, and the two strongest word embedding-based methods, DistilBERT minimizing Reconstruction loss and RoBERTa maximizing KL Divergence. Below we present the full table of

	Method	50	100	250	500	750	1000	1500	2000	2500	3000
1	BERT, Cosine	0.535 (0.011)	0.554 (0.013)	0.591 (0.011)	0.613 (0.011)	0.645 (0.009)	0.65 (0.009)	0.669 (0.006)	0.671 (0.007)	0.672 (0.008)	0.68 (0.008)
2	BERT, d-Optimality	0.534 (0.012)	0.54 (0.006)	0.573 (0.015)	0.599 (0.011)	0.62 (0.014)	0.629 (0.009)	0.639 (0.011)	0.657 (0.009)	0.656 (0.009)	0.668 (0.009)
3	BERT, K-L Divergence	0.536 (0.008)	0.557 (0.009)	0.586 (0.015)	0.617 (0.013)	0.637 (0.01)	0.645 (0.009)	0.659 (0.006)	0.67 (0.006)	0.685 (0.004)	0.688 (0.004)
4	BERT, k-Means	0.531 (0.013)	0.549 (0.011)	0.584 (0.015)	0.625 (0.006)	0.637 (0.008)	0.645 (0.007)	0.671 (0.004)	0.668 (0.006)	0.678 (0.008)	0.673 (0.008)
5	BERT, K-S Distance	0.547 (0.013)	0.554 (0.014)	0.593 (0.013)	0.614 (0.011)	0.625 (0.011)	0.642 (0.011)	0.659 (0.011)	0.669 (0.011)	0.679 (0.009)	0.691 (0.005)
6	BERT, Reconstruction Loss	0.537 (0.011)	0.554 (0.012)	0.605 (0.008)	0.624 (0.01)	0.639 (0.011)	0.651 (0.011)	0.662 (0.008)	0.672 (0.007)	0.677 (0.006)	0.682 (0.005)
7	BoW + NMF, Cosine	0.522 (0.008)	0.545 (0.007)	0.578 (0.01)	0.609 (0.008)	0.595 (0.008)	0.606 (0.01)	0.615 (0.009)	0.61 (0.01)	0.611 (0.007)	0.631 (0.008)
8	BoW + NMF, d-Optimality	0.531 (0.01)	0.555 (0.01)	0.588 (0.01)	0.631 (0.008)	0.646 (0.004)	0.648 (0.005)	0.666 (0.004)	0.675 (0.004)	0.681 (0.004)	0.689 (0.004)
9	BoW + NMF, K-L Divergence	0.543 (0.008)	0.57 (0.011)	0.585 (0.012)	0.601 (0.011)	0.61 (0.009)	0.618 (0.011)	0.64 (0.008)	0.646 (0.008)	0.661 (0.009)	0.668 (0.007)
10	BoW + NMF, k-Means	0.539 (0.012)	0.565 (0.011)	0.58 (0.011)	0.624 (0.006)	0.623 (0.006)	0.655 (0.006)	0.651 (0.009)	0.678 (0.005)	0.671 (0.007)	0.671 (0.006)
11	BoW + NMF, K-S Distance	0.543 (0.011)	0.557 (0.012)	0.574 (0.015)	0.573 (0.013)	0.562 (0.013)	0.554 (0.01)	0.571 (0.011)	0.649 (0.006)	0.648 (0.008)	0.62 (0.007)
12	BoW + NMF, Reconstruction Loss	0.524 (0.007)	0.566 (0.012)	0.593 (0.013)	0.614 (0.011)	0.628 (0.011)	0.643 (0.009)	0.66 (0.007)	0.678 (0.004)	0.684 (0.004)	0.685 (0.005)
13	BoW + PCA, Cosine	0.52 (0.01)	0.527 (0.011)	0.561 (0.008)	0.591 (0.013)	0.593 (0.011)	0.616 (0.012)	0.635 (0.01)	0.643 (0.01)	0.656 (0.008)	0.656 (0.008)
14	BoW + PCA, d-Optimality	0.529 (0.01)	0.551 (0.012)	0.592 (0.011)	0.621 (0.007)	0.644 (0.008)	0.646 (0.004)	0.666 (0.005)	0.677 (0.005)	0.682 (0.003)	0.692 (0.003)
15	BoW + PCA, K-L Divergence	0.541 (0.016)	0.547 (0.011)	0.583 (0.014)	0.61 (0.013)	0.627 (0.01)	0.637 (0.007)	0.651 (0.008)	0.663 (0.006)	0.672 (0.007)	0.676 (0.005)
16	BoW + PCA, k-Means	0.548 (0.012)	0.555 (0.011)	0.586 (0.009)	0.606 (0.01)	0.641 (0.004)	0.639 (0.008)	0.654 (0.008)	0.669 (0.008)	0.672 (0.007)	0.679 (0.006)
17	BoW + PCA, K-S Distance	0.534 (0.011)	0.556 (0.015)	0.598 (0.011)	0.6 (0.012)	0.578 (0.01)	0.562 (0.007)	0.563 (0.009)	0.555 (0.01)	0.549 (0.008)	0.538 (0.007)
18	BoW + PCA, Reconstruction Loss	0.539 (0.011)	0.552 (0.01)	0.587 (0.013)	0.614 (0.009)	0.642 (0.006)	0.656 (0.007)	0.67 (0.008)	0.676 (0.007)	0.684 (0.006)	0.689 (0.006)
19	BoW + tSNE, Cosine	0.535 (0.014)	0.526 (0.01)	0.561 (0.007)	0.613 (0.008)	0.625 (0.009)	0.633 (0.008)	0.646 (0.01)	0.66 (0.01)	0.674 (0.007)	0.68 (0.006)
20	BoW + tSNE, d-Optimality	0.512 (0.008)	0.542 (0.009)	0.577 (0.01)	0.604 (0.006)	0.623 (0.005)	0.634 (0.006)	0.657 (0.005)	0.672 (0.005)	0.678 (0.005)	0.69 (0.005)
21	BoW + tSNE, K-L Divergence	0.539 (0.013)	0.555 (0.011)	0.58 (0.009)	0.606 (0.011)	0.611 (0.01)	0.618 (0.008)	0.644 (0.01)	0.663 (0.007)	0.671 (0.005)	0.677 (0.006)
22	BoW + tSNE, k-Means	0.543 (0.01)	0.554 (0.013)	0.576 (0.01)	0.61 (0.01)	0.638 (0.007)	0.632 (0.012)	0.644 (0.007)	0.671 (0.009)	0.673 (0.007)	0.669 (0.008)
23	BoW + tSNE, K-S Distance	0.559 (0.01)	0.569 (0.007)	0.561 (0.006)	0.574 (0.014)	0.597 (0.005)	0.593 (0.016)	0.573 (0.015)	0.527 (0.013)	0.517 (0.007)	0.519 (0.005)
24	BoW + tSNE, Reconstruction Loss	0.526 (0.009)	0.53 (0.008)	0.569 (0.005)	0.608 (0.008)	0.623 (0.005)	0.646 (0.006)	0.663 (0.005)	0.67 (0.005)	0.681 (0.005)	0.693 (0.005)
25	BoW + UMAP, Cosine	0.532 (0.012)	0.544 (0.009)	0.571 (0.01)	0.615 (0.008)	0.64 (0.006)	0.643 (0.008)	0.657 (0.007)	0.666 (0.007)	0.671 (0.008)	0.67 (0.007)
26	BoW + UMAP, d-Optimality	0.52 (0.01)	0.541 (0.008)	0.565 (0.008)	0.605 (0.009)	0.621 (0.008)	0.643 (0.005)	0.658 (0.005)	0.672 (0.005)	0.681 (0.005)	0.684 (0.006)
27	BoW + UMAP, K-L Divergence	0.537 (0.008)	0.564 (0.013)	0.574 (0.013)	0.602 (0.006)	0.625 (0.009)	0.646 (0.008)	0.654 (0.007)	0.662 (0.008)	0.667 (0.008)	0.672 (0.008)
28	BoW + UMAP, k-Means	0.561 (0.012)	0.555 (0.014)	0.575 (0.01)	0.609 (0.012)	0.61 (0.013)	0.646 (0.007)	0.654 (0.005)	0.663 (0.008)	0.664 (0.006)	0.677 (0.007)
29	BoW + UMAP, K-S Distance	0.537 (0.012)	0.54 (0.007)	0.54 (0.009)	0.578 (0.009)	0.599 (0.009)	0.6 (0.011)	0.581 (0.008)	0.57 (0.006)	0.561 (0.009)	0.557 (0.007)
30	BoW + UMAP, Reconstruction Loss	0.53 (0.008)	0.545 (0.007)	0.567 (0.011)	0.623 (0.011)	0.634 (0.01)	0.639 (0.01)	0.655 (0.009)	0.674 (0.008)	0.684 (0.006)	0.684 (0.006)
31	DistilBERT, Cosine	0.518 (0.012)	0.527 (0.01)	0.554 (0.01)	0.588 (0.013)	0.609 (0.012)	0.619 (0.01)	0.634 (0.01)	0.649 (0.009)	0.658 (0.009)	0.667 (0.009)
32	DistilBERT, d-Optimality	0.514 (0.011)	0.547 (0.009)	0.582 (0.014)	0.606 (0.012)	0.625 (0.008)	0.624 (0.012)	0.647 (0.009)	0.655 (0.01)	0.662 (0.007)	0.667 (0.007)
33	DistilBERT, K-L Divergence	0.533 (0.017)	0.556 (0.014)	0.563 (0.013)	0.606 (0.011)	0.63 (0.009)	0.637 (0.01)	0.658 (0.005)	0.669 (0.005)	0.681 (0.003)	0.688 (0.004)
34	DistilBERT, k-Means	0.543 (0.01)	0.548 (0.011)	0.581 (0.009)	0.613 (0.004)	0.621 (0.01)	0.637 (0.008)	0.652 (0.007)	0.666 (0.006)	0.667 (0.006)	0.682 (0.009)
35	DistilBERT, K-S Distance	0.524 (0.011)	0.543 (0.011)	0.581 (0.011)	0.606 (0.007)	0.64 (0.007)	0.645 (0.008)	0.662 (0.005)	0.67 (0.005)	0.678 (0.006)	0.68 (0.006)
36	GLoVE, Cosine	0.534 (0.013)	0.552 (0.012)	0.571 (0.012)	0.594 (0.014)	0.621 (0.012)	0.631 (0.011)	0.642 (0.01)	0.649 (0.009)	0.65 (0.01)	0.658 (0.009)
37	GLoVE, d-Optimality	0.526 (0.01)	0.558 (0.008)	0.602 (0.01)	0.606 (0.011)	0.624 (0.011)	0.636 (0.011)	0.655 (0.006)	0.667 (0.008)	0.672 (0.006)	0.675 (0.006)
38	GLoVE, K-L Divergence	0.537 (0.007)	0.552 (0.009)	0.579 (0.01)	0.619 (0.01)	0.635 (0.011)	0.646 (0.008)	0.662 (0.009)	0.666 (0.007)	0.678 (0.008)	0.685 (0.007)
39	GLoVE, k-Means	0.528 (0.01)	0.553 (0.013)	0.593 (0.009)	0.605 (0.011)	0.636 (0.008)	0.643 (0.009)	0.646 (0.006)	0.671 (0.007)	0.672 (0.005)	0.68 (0.004)
40	GLoVE, K-S Distance	0.545 (0.01)	0.565 (0.012)	0.597 (0.011)	0.616 (0.013)	0.637 (0.004)	0.647 (0.005)	0.665 (0.006)	0.668 (0.008)	0.671 (0.009)	0.675 (0.008)
41	LDA, Cosine	0.527 (0.008)	0.556 (0.007)	0.584 (0.011)	0.622 (0.006)	0.643 (0.005)	0.65 (0.008)	0.661 (0.008)	0.67 (0.007)	0.68 (0.006)	0.689 (0.004)
42	LDA, d-Optimality	0.556 (0.01)	0.545 (0.009)	0.567 (0.011)	0.6 (0.005)	0.623 (0.006)	0.639 (0.009)	0.649 (0.011)	0.671 (0.006)	0.674 (0.004)	0.679 (0.005)
43	LDA, K-L Divergence	0.547 (0.012)	0.556 (0.01)	0.585 (0.008)	0.612 (0.009)	0.631 (0.009)	0.638 (0.007)	0.647 (0.01)	0.662 (0.009)	0.673 (0.007)	0.681 (0.007)
44	LDA, k-Means	0.534 (0.013)	0.564 (0.014)	0.572 (0.008)	0.587 (0.012)	0.643 (0.005)	0.644 (0.01)	0.665 (0.007)	0.667 (0.006)	0.678 (0.007)	0.69 (0.006)
45	LDA, K-S Distance	0.531 (0.01)	0.567 (0.008)	0.597 (0.011)	0.625 (0.008)	0.637 (0.007)	0.655 (0.004)	0.655 (0.007)	0.669 (0.004)	0.679 (0.004)	0.683 (0.004)
46	Random, Random	0.546 (0.01)	0.554 (0.011)	0.603 (0.01)	0.621 (0.007)	0.642 (0.008)	0.645 (0.004)	0.656 (0.006)	0.664 (0.005)	0.663 (0.006)	0.667 (0.005)
47	RoBERTa, Cosine	0.523 (0.008)	0.553 (0.007)	0.59 (0.013)	0.612 (0.012)	0.621 (0.01)	0.63 (0.01)	0.637 (0.013)	0.653 (0.013)	0.667 (0.009)	0.673 (0.01)
48	RoBERTa, d-Optimality	0.527 (0.014)	0.561 (0.011)	0.575 (0.01)	0.607 (0.008)	0.621 (0.012)	0.641 (0.01)	0.645 (0.009)	0.669 (0.008)	0.667 (0.011)	0.676 (0.008)
49	RoBERTa, K-L Divergence	0.55 (0.013)	0.555 (0.014)	0.581 (0.011)	0.621 (0.008)	0.64 (0.008)	0.644 (0.007)	0.661 (0.007)	0.674 (0.004)	0.68 (0.005)	0.687 (0.004)
50	RoBERTa, k-Means	0.53 (0.01)	0.565 (0.01)	0.579 (0.009)	0.615 (0.009)	0.637 (0.007)	0.642 (0.008)	0.668 (0.008)	0.659 (0.011)	0.678 (0.004)	0.684 (0.005)
51	RoBERTa, K-S Distance	0.536 (0.011)	0.566 (0.008)	0.58 (0.009)	0.609 (0.011)	0.622 (0.011)	0.629 (0.011)	0.652 (0.009)	0.663 (0.009)	0.667 (0.012)	0.68 (0.008)
52	RoBERTa, Reconstruction Loss	0.533 (0.014)	0.555 (0.012)	0.601 (0.009)	0.612 (0.01)	0.64 (0.006)	0.643 (0.007)	0.65 (0.01)	0.668 (0.01)	0.672 (0.01)	0.678 (0.01)
53	Universal Encoder, Cosine	0.554 (0.012)	0.57 (0.011)	0.589 (0.011)	0.608 (0.011)	0.615 (0.01)	0.636 (0.006)	0.649 (0.006)	0.642 (0.01)	0.655 (0.009)	0.67 (0.008)
54	Universal Encoder, d-Optimality	0.542 (0.006)	0.534 (0.009)	0.557 (0.011)	0.61 (0.01)	0.633 (0.009)	0.647 (0.01)	0.644 (0.008)	0.661 (0.008)	0.665 (0.007)	0.679 (0.007)
55	Universal Encoder, K-L Divergence	0.526 (0.008)	0.556 (0.009)	0.585 (0.008)	0.598 (0.011)	0.607 (0.012)	0.64 (0.006)	0.657 (0.009)	0.679 (0.005)	0.685 (0.006)	0.688 (0.005)
56	Universal Encoder, k-Means	0.546 (0.012)	0.567 (0.009)	0.584 (0.008)	0.619 (0.009)	0.627 (0.01)	0.647 (0.007)	0.656 (0.007)	0.662 (0.007)	0.678 (0.004)	0.675 (0.007)
57	Universal Encoder, K-S Distance	0.522 (0.009)	0.559 (0.011)	0.595 (0.012)	0.612 (0.009)	0.62 (0.007)	0.638 (0.01)	0.647 (0.007)	0.66 (0.008)	0.668 (0.005)	0.674 (0.007)

Table 1: Full AUC results for all methods in the Executive Orders case.

all methods and their AUCs for each k and for both applications; we also summarize the tables, showing which representations and which distance metrics perform best in general.

As well, we can calculate whether the methods that perform well in one application also tend to perform well in the other. To this effect, we calculate Spearman's rank order correlation between each method's AUC across applications, and find that it is 0.88. This gives us confidence that which methods perform best may be robust to application choice.

In most cases, we observe that the random sampling method, the one used by nearly every applied researcher, systematically underperforms. The selection methods that generally perform the best are k-means clustering, maximizing KL divergence, or minimizing reconstruction error.

A related analysis in Table 3 examines the relationship between AUC and each distance metric and representation method via regression analysis. Each unit is a single simulation: since we repeat each combination of k, distance metric, and representation 10 times to generate confidence intervals, we have 10 observations per combination in this regression. The representation base category is

	Method	50	100	250	500	750	1000	1500	2000	2500	3000
1	BERT, Cosine	0.562 (0.017)	0.59 (0.014)	0.661 (0.012)	0.685 (0.013)	0.719 (0.01)	0.724 (0.01)	0.737 (0.008)	0.742 (0.007)	0.747 (0.007)	0.749 (0.008)
2	BERT, d-Optimality	0.541 (0.011)	0.591 (0.019)	0.623 (0.017)	0.692 (0.01)	0.699 (0.008)	0.73 (0.01)	0.738 (0.009)	0.743 (0.007)	0.748 (0.006)	0.751 (0.007)
3	BERT, K-L Divergence	0.548 (0.013)	0.581 (0.019)	0.664 (0.008)	0.696 (0.008)	0.706 (0.009)	0.721 (0.009)	0.728 (0.008)	0.741 (0.008)	0.749 (0.008)	0.749 (0.009)
4	BERT, k-Means	0.551 (0.017)	0.576 (0.015)	0.655 (0.01)	0.68 (0.008)	0.705 (0.012)	0.719 (0.007)	0.727 (0.007)	0.737 (0.009)	0.743 (0.008)	0.743 (0.008)
5	BERT, K-S Distance	0.544 (0.016)	0.572 (0.015)	0.623 (0.014)	0.675 (0.011)	0.709 (0.012)	0.719 (0.01)	0.737 (0.009)	0.745 (0.007)	0.747 (0.007)	0.748 (0.007)
6	BERT, Reconstruction Loss	0.557 (0.013)	0.589 (0.022)	0.616 (0.017)	0.667 (0.017)	0.694 (0.008)	0.713 (0.011)	0.732 (0.007)	0.742 (0.007)	0.743 (0.006)	0.748 (0.007)
7	BoW + NMF, Cosine	0.541 (0.013)	0.59 (0.019)	0.665 (0.011)	0.689 (0.009)	0.706 (0.009)	0.721 (0.009)	0.74 (0.008)	0.746 (0.008)	0.747 (0.007)	0.749 (0.007)
8	BoW + NMF, d-Optimality	0.542 (0.011)	0.576 (0.019)	0.65 (0.013)	0.7 (0.006)	0.715 (0.006)	0.72 (0.01)	0.738 (0.008)	0.745 (0.009)	0.746 (0.007)	0.748 (0.007)
9	BoW + NMF, K-L Divergence	0.544 (0.005)	0.611 (0.005)	0.685 (0.008)	0.699 (0.006)	0.681 (0.007)	0.72 (0.007)	0.727 (0.007)	0.728 (0.006)	0.727 (0.006)	0.738 (0.007)
10	BoW + NMF, k-Means	0.556 (0.014)	0.598 (0.018)	0.626 (0.017)	0.694 (0.007)	0.707 (0.009)	0.714 (0.011)	0.728 (0.006)	0.738 (0.007)	0.742 (0.009)	0.746 (0.007)
11	BoW + NMF, K-S Distance	0.548 (0.011)	0.575 (0.012)	0.622 (0.012)	0.665 (0.011)	0.704 (0.01)	0.721 (0.007)	0.732 (0.008)	0.739 (0.009)	0.743 (0.009)	0.75 (0.008)
12	BoW + NMF, Reconstruction Loss	0.53 (0.013)	0.551 (0.008)	0.632 (0.011)	0.675 (0.012)	0.696 (0.009)	0.711 (0.009)	0.728 (0.006)	0.737 (0.005)	0.744 (0.007)	0.746 (0.007)
13	BoW + PCA, Cosine	0.563 (0.01)	0.582 (0.012)	0.645 (0.013)	0.683 (0.008)	0.715 (0.008)	0.731 (0.009)	0.739 (0.008)	0.741 (0.007)	0.744 (0.006)	0.749 (0.007)
14	BoW + PCA, d-Optimality	0.554 (0.013)	0.573 (0.01)	0.641 (0.012)	0.691 (0.005)	0.703 (0.011)	0.719 (0.01)	0.734 (0.009)	0.742 (0.008)	0.745 (0.008)	0.746 (0.007)
15	BoW + PCA, K-L Divergence	0.549 (0.015)	0.572 (0.016)	0.606 (0.022)	0.672 (0.012)	0.7 (0.012)	0.719 (0.01)	0.737 (0.01)	0.742 (0.01)	0.744 (0.008)	0.749 (0.007)
16	BoW + PCA, k-Means	0.562 (0.019)	0.589 (0.009)	0.637 (0.019)	0.701 (0.008)	0.71 (0.008)	0.723 (0.009)	0.734 (0.006)	0.736 (0.007)	0.745 (0.007)	0.747 (0.008)
17	BoW + PCA, K-S Distance	0.552 (0.013)	0.597 (0.013)	0.654 (0.008)	0.681 (0.01)	0.706 (0.009)	0.715 (0.007)	0.726 (0.007)	0.739 (0.009)	0.747 (0.008)	0.749 (0.007)
18	BoW + PCA, Reconstruction Loss	0.586 (0.016)	0.592 (0.009)	0.651 (0.009)	0.685 (0.01)	0.711 (0.009)	0.727 (0.007)	0.734 (0.007)	0.746 (0.007)	0.749 (0.007)	0.75 (0.007)
19	BoW + tSNE, Cosine	0.531 (0.01)	0.573 (0.017)	0.633 (0.017)	0.695 (0.007)	0.705 (0.006)	0.715 (0.009)	0.727 (0.009)	0.738 (0.009)	0.742 (0.009)	0.746 (0.008)
20	BoW + tSNE, d-Optimality	0.575 (0.011)	0.61 (0.01)	0.656 (0.011)	0.693 (0.012)	0.712 (0.009)	0.719 (0.007)	0.732 (0.008)	0.737 (0.008)	0.741 (0.009)	0.743 (0.008)
21	BoW + tSNE, K-L Divergence	0.572 (0.009)	0.614 (0.007)	0.646 (0.011)	0.683 (0.009)	0.698 (0.011)	0.71 (0.01)	0.732 (0.009)	0.738 (0.008)	0.743 (0.008)	0.745 (0.008)
22	BoW + tSNE, k-Means	0.585 (0.015)	0.606 (0.014)	0.627 (0.018)	0.679 (0.008)	0.706 (0.01)	0.715 (0.008)	0.729 (0.01)	0.738 (0.008)	0.743 (0.007)	0.751 (0.008)
23	BoW + tSNE, K-S Distance	0.529 (0.012)	0.534 (0.013)	0.607 (0.014)	0.636 (0.014)	0.677 (0.009)	0.71 (0.004)	0.72 (0.006)	0.738 (0.007)	0.74 (0.007)	0.74 (0.006)
24	BoW + tSNE, Reconstruction Loss	0.572 (0.015)	0.617 (0.012)	0.663 (0.01)	0.692 (0.005)	0.715 (0.006)	0.721 (0.007)	0.734 (0.007)	0.742 (0.009)	0.744 (0.009)	0.747 (0.009)
25	BoW + UMAP, Cosine	0.548 (0.011)	0.578 (0.01)	0.651 (0.015)	0.686 (0.008)	0.716 (0.009)	0.729 (0.007)	0.74 (0.007)	0.74 (0.007)	0.744 (0.008)	0.747 (0.008)
26	BoW + UMAP, d-Optimality	0.573 (0.014)	0.61 (0.019)	0.676 (0.01)	0.697 (0.006)	0.715 (0.007)	0.723 (0.008)	0.734 (0.007)	0.738 (0.008)	0.74 (0.008)	0.745 (0.007)
27	BoW + UMAP, K-L Divergence	0.554 (0.011)	0.589 (0.016)	0.639 (0.021)	0.686 (0.011)	0.705 (0.006)	0.719 (0.007)	0.725 (0.007)	0.735 (0.007)	0.741 (0.007)	0.744 (0.007)
28	BoW + UMAP, k-Means	0.557 (0.015)	0.591 (0.019)	0.649 (0.016)	0.701 (0.01)	0.697 (0.012)	0.73 (0.008)	0.725 (0.007)	0.74 (0.006)	0.743 (0.007)	0.742 (0.008)
29	BoW + UMAP, K-S Distance	0.558 (0.01)	0.576 (0.012)	0.644 (0.01)	0.655 (0.011)	0.707 (0.008)	0.719 (0.008)	0.721 (0.008)	0.738 (0.007)	0.741 (0.007)	0.737 (0.007)
30	BoW + UMAP, Reconstruction Loss	0.544 (0.009)	0.579 (0.014)	0.64 (0.013)	0.66 (0.012)	0.685 (0.009)	0.712 (0.007)	0.729 (0.007)	0.737 (0.007)	0.742 (0.007)	0.746 (0.006)
31	DistilBERT, Cosine	0.547 (0.011)	0.593 (0.016)	0.653 (0.02)	0.696 (0.014)	0.71 (0.008)	0.727 (0.008)	0.739 (0.006)	0.743 (0.006)	0.748 (0.006)	0.751 (0.006)
32	DistilBERT, d-Optimality	0.549 (0.015)	0.581 (0.015)	0.659 (0.011)	0.684 (0.01)	0.712 (0.006)	0.719 (0.01)	0.738 (0.006)	0.739 (0.008)	0.745 (0.007)	0.749 (0.008)
33	DistilBERT, K-L Divergence	0.573 (0.02)	0.575 (0.022)	0.648 (0.01)	0.695 (0.008)	0.709 (0.006)	0.721 (0.004)	0.734 (0.004)	0.741 (0.005)	0.746 (0.005)	0.749 (0.006)
34	DistilBERT, k-Means	0.567 (0.012)	0.585 (0.017)	0.63 (0.012)	0.683 (0.007)	0.706 (0.012)	0.714 (0.005)	0.731 (0.008)	0.743 (0.007)	0.744 (0.007)	0.746 (0.007)
35	DistilBERT, K-S Distance	0.573 (0.014)	0.605 (0.015)	0.643 (0.013)	0.673 (0.011)	0.713 (0.008)	0.717 (0.007)	0.732 (0.006)	0.737 (0.008)	0.745 (0.009)	0.743 (0.008)
36	GLoVE, Cosine	0.569 (0.012)	0.61 (0.014)	0.657 (0.014)	0.691 (0.009)	0.704 (0.01)	0.72 (0.005)	0.735 (0.006)	0.742 (0.007)	0.745 (0.007)	0.749 (0.007)
37	GLoVE, d-Optimality	0.553 (0.018)	0.593 (0.011)	0.662 (0.016)	0.7 (0.011)	0.706 (0.008)	0.718 (0.008)	0.731 (0.007)	0.737 (0.007)	0.742 (0.007)	0.745 (0.007)
38	GLoVE, K-L Divergence	0.544 (0.005)	0.611 (0.005)	0.685 (0.008)	0.699 (0.006)	0.681 (0.007)	0.72 (0.007)	0.727 (0.007)	0.728 (0.006)	0.727 (0.006)	0.738 (0.007)
39	GLoVE, k-Means	0.505 (0.058)	0.578 (0.017)	0.649 (0.011)	0.686 (0.009)	0.707 (0.008)	0.718 (0.006)	0.739 (0.009)	0.738 (0.008)	0.746 (0.007)	0.75 (0.008)
40	GLoVE, K-S Distance	0.554 (0.014)	0.581 (0.009)	0.644 (0.012)	0.665 (0.013)	0.713 (0.005)	0.723 (0.006)	0.733 (0.007)	0.738 (0.007)	0.741 (0.008)	0.747 (0.008)
41	LDA, Cosine	0.557 (0.014)	0.596 (0.015)	0.665 (0.009)	0.694 (0.009)	0.704 (0.008)	0.721 (0.008)	0.731 (0.007)	0.74 (0.008)	0.746 (0.008)	0.748 (0.007)
42	LDA, d-Optimality	0.566 (0.016)	0.589 (0.015)	0.638 (0.015)	0.672 (0.012)	0.693 (0.01)	0.704 (0.011)	0.726 (0.009)	0.738 (0.008)	0.743 (0.008)	0.748 (0.008)
43	LDA, K-L Divergence	0.554 (0.01)	0.579 (0.013)	0.624 (0.012)	0.679 (0.015)	0.708 (0.008)	0.724 (0.01)	0.732 (0.01)	0.738 (0.009)	0.745 (0.007)	0.747 (0.007)
44	LDA, k-Means	0.562 (0.015)	0.605 (0.012)	0.64 (0.014)	0.678 (0.016)	0.712 (0.01)	0.723 (0.007)	0.739 (0.005)	0.74 (0.007)	0.742 (0.011)	0.746 (0.007)
45	LDA, K-S Distance	0.556 (0.008)	0.604 (0.009)	0.655 (0.015)	0.69 (0.013)	0.709 (0.01)	0.718 (0.01)	0.727 (0.008)	0.734 (0.008)	0.739 (0.007)	0.744 (0.007)
46	Random, Random	0.563 (0.015)	0.573 (0.022)	0.635 (0.02)	0.675 (0.011)	0.699 (0.009)	0.703 (0.008)	0.724 (0.007)	0.724 (0.007)	0.732 (0.007)	0.735 (0.008)
47	RoBERTa, Cosine	0.559 (0.015)	0.606 (0.013)	0.64 (0.02)	0.682 (0.013)	0.708 (0.008)	0.722 (0.006)	0.734 (0.005)	0.74 (0.005)	0.741 (0.005)	0.747 (0.006)
48	RoBERTa, d-Optimality	0.583 (0.014)	0.568 (0.018)	0.649 (0.018)	0.685 (0.014)	0.701 (0.013)	0.721 (0.007)	0.729 (0.008)	0.74 (0.007)	0.745 (0.007)	0.752 (0.007)
49	RoBERTa, K-L Divergence	0.55 (0.014)	0.613 (0.009)	0.652 (0.016)	0.691 (0.01)	0.716 (0.008)	0.717 (0.01)	0.732 (0.008)	0.737 (0.008)	0.739 (0.007)	0.744 (0.008)
50	RoBERTa, k-Means	0.561 (0.015)	0.574 (0.016)	0.639 (0.017)	0.69 (0.009)	0.701 (0.005)	0.717 (0.006)	0.735 (0.006)	0.741 (0.007)	0.745 (0.008)	0.749 (0.006)
51	RoBERTa, K-S Distance	0.547 (0.015)	0.579 (0.02)	0.621 (0.013)	0.674 (0.01)	0.69 (0.01)	0.712 (0.008)	0.731 (0.005)	0.737 (0.007)	0.74 (0.006)	0.743 (0.006)
52	RoBERTa, Reconstruction Loss	0.543 (0.017)	0.567 (0.014)	0.627 (0.015)	0.691 (0.01)	0.709 (0.009)	0.722 (0.007)	0.735 (0.008)	0.742 (0.006)	0.744 (0.007)	0.746 (0.008)
53	Universal Encoder, Cosine	0.553 (0.01)	0.571 (0.014)	0.649 (0.012)	0.699 (0.007)	0.725 (0.006)	0.732 (0.007)	0.74 (0.007)	0.742 (0.006)	0.744 (0.008)	0.748 (0.008)
54	Universal Encoder, d-Optimality	0.566 (0.015)	0.57 (0.015)	0.657 (0.013)	0.698 (0.006)	0.715 (0.009)	0.726 (0.008)	0.741 (0.006)	0.747 (0.008)	0.744 (0.007)	0.748 (0.007)
55	Universal Encoder, K-L Divergence	0.534 (0.013)	0.558 (0.015)	0.603 (0.02)	0.675 (0.013)	0.699 (0.009)	0.718 (0.009)	0.729 (0.009)	0.742 (0.01)	0.746 (0.008)	0.748 (0.008)
56	Universal Encoder, k-Means	0.555 (0.013)	0.577 (0.012)	0.618 (0.015)	0.693 (0.011)	0.705 (0.011)	0.716 (0.01)	0.732 (0.007)	0.743 (0.006)	0.745 (0.008)	0.75 (0.008)
57	Universal Encoder, K-S Distance	0.56 (0.007)	0.575 (0.009)	0.64 (0.01)	0.673 (0.011)	0.703 (0.006)	0.713 (0.006)	0.726 (0.004)	0.735 (0.005)	0.742 (0.007)	0.743 (0.007)

Table 2: Full AUC results for all methods in the StockTwits case.

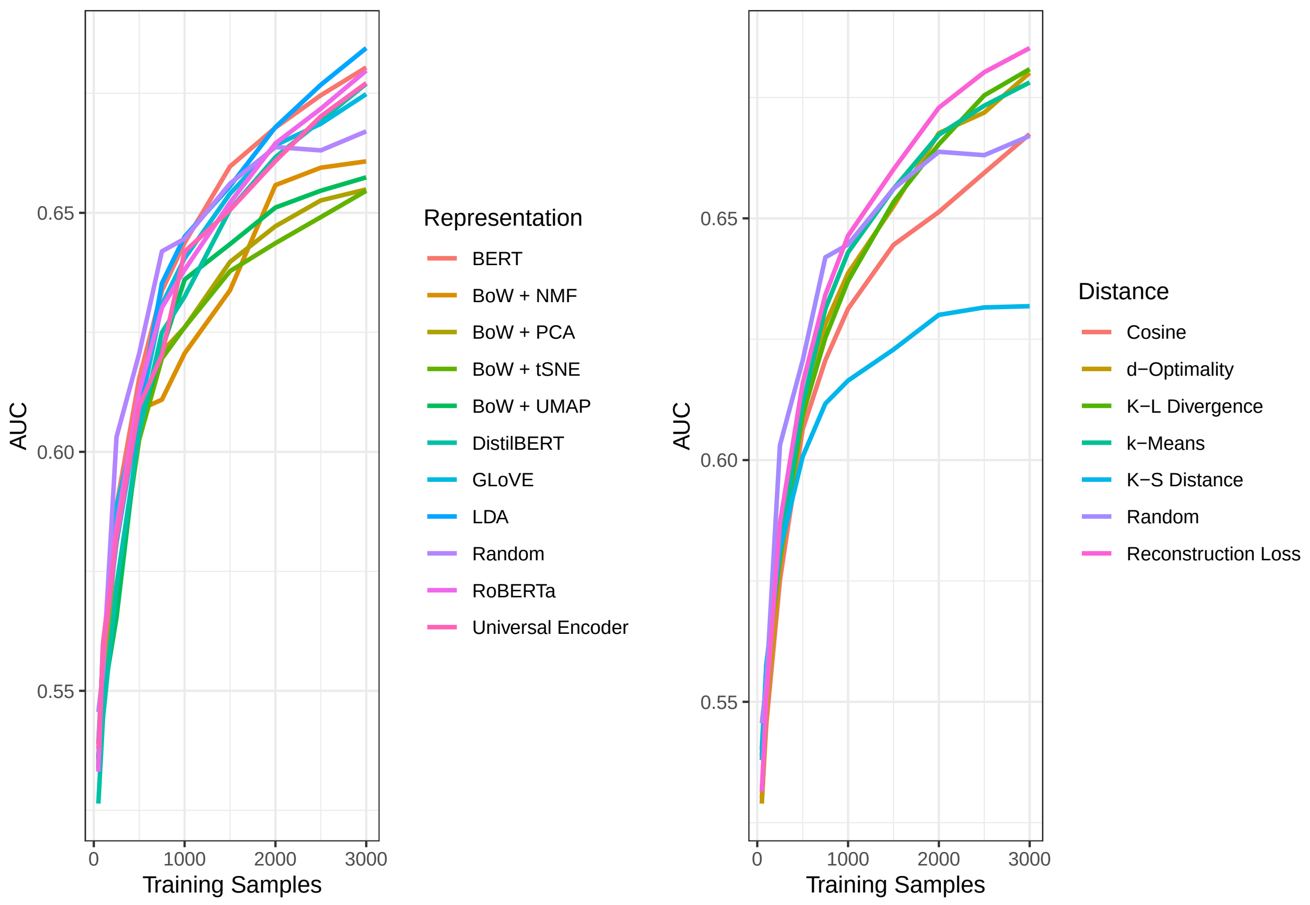

Figure 1: Average AUCs, for each value of k, for each representation and distance metric, in the StockTwits application.

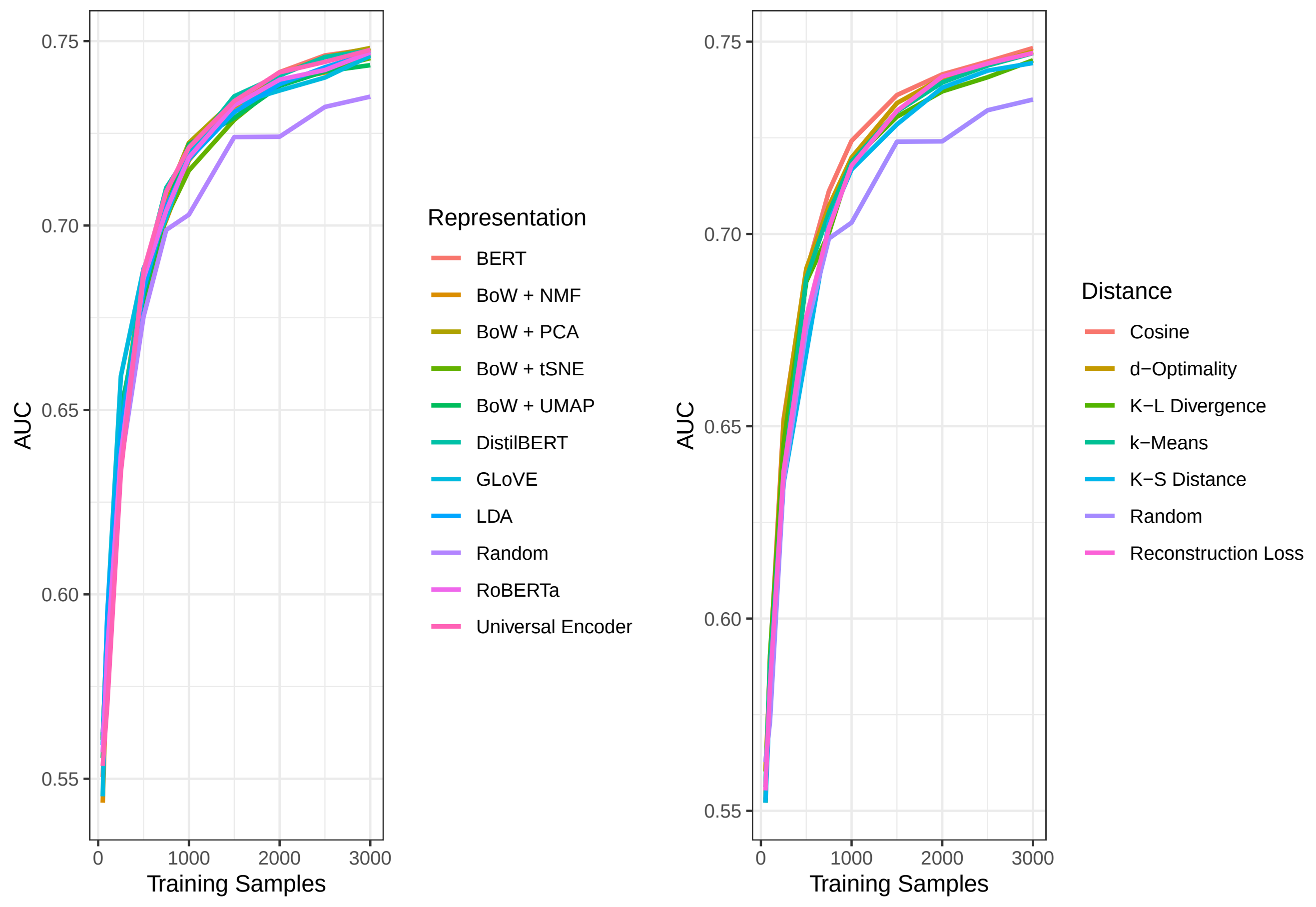

Figure 2: Average AUCs, for each value of k, for each representation and distance metric, in the Executive Orders application.

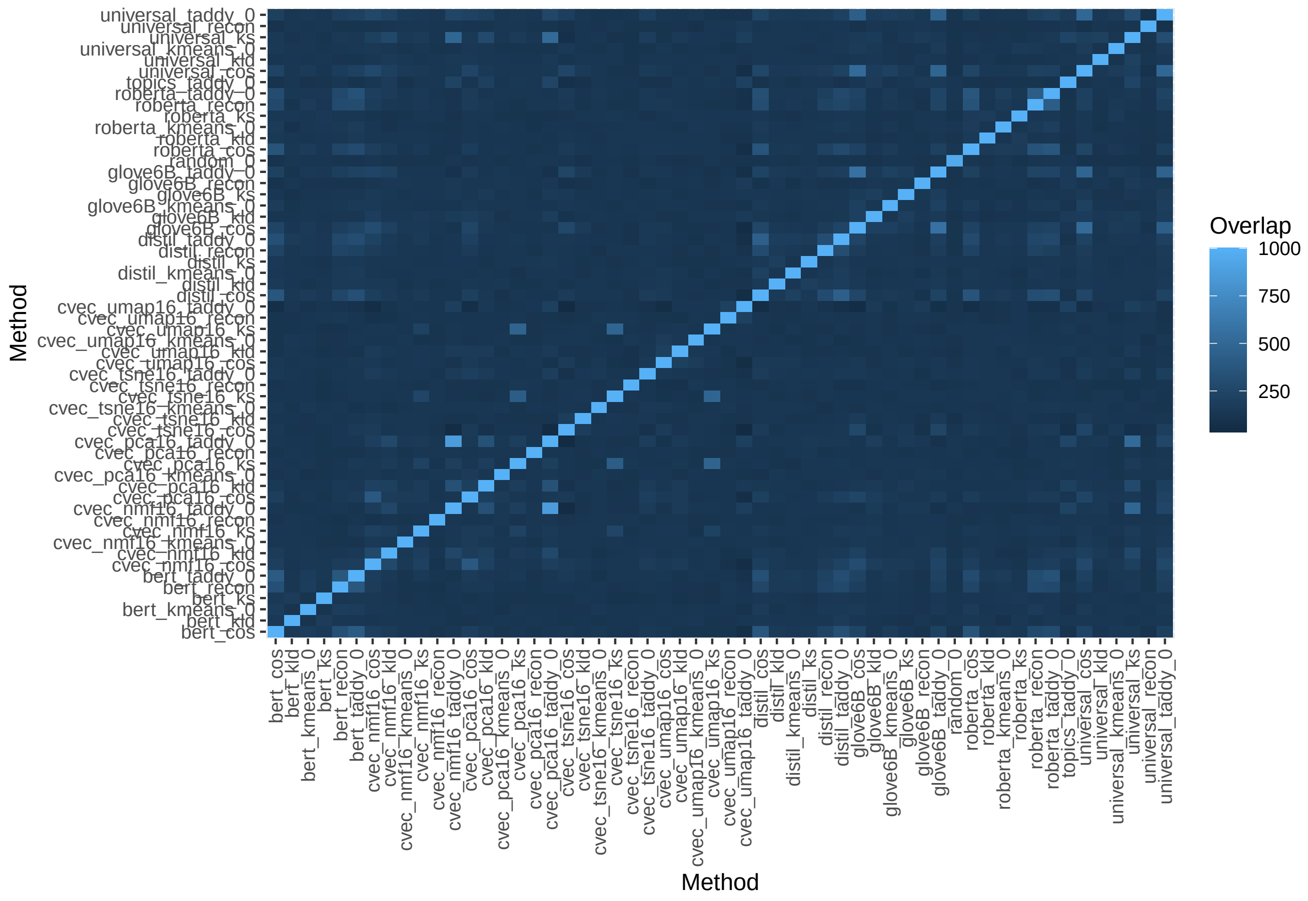

Figure 3: After selecting 1000 training observations in the Executive Orders application, the average number of in-common observations between any two methods is 140 and the median is 126.

BERT, and the distance metric base category is cosine distance. Generally, we find that BERT outperforms the bag-of-words methods and performs comparably to the other embedding methods; the best-performing distance metric is Reconstruction loss.

This gives us confidence that, even though there is significant variation in accuracy across methods, BERT appears to be a high performer in the Executive Orders application (though not the StockTwits application).

3.1 Training Set Overlap

Examining the Executive Orders application, we observe that very few of the methods produce any significant overlap with any other, indicating that they are all producing meaningfully different training sets.

	Dependent variable:	
	AUC	
	(1) Executive Orders	(2) StockTwits
Representation: BoW + NMF	−0.012	−0.001
	(0.003)	(0.004)
Representation: BoW + PCA	−0.014	0.002
	(0.003)	(0.004)
Representation: BoW + tSNE	−0.017	−0.001
	(0.003)	(0.004)
Representation: BoW + UMAP	−0.013	−0.00001
	(0.003)	(0.004)
Representation: DistilBERT	−0.007	0.003
	(0.003)	(0.004)
Representation: GLoVE	−0.001	0.001
	(0.003)	(0.004)
Representation: LDA	0.002	0.001
	(0.003)	(0.004)
Representation: Random	0.006	−0.012
	(0.006)	(0.008)
Representation: RoBERTa	−0.003	−0.0003
	(0.003)	(0.004)
Representation: Universal Encoder	−0.002	−0.001
	(0.003)	(0.004)
Distance: d-Optimality	0.007	−0.001
	(0.003)	(0.003)
Distance: KL Divergence	0.009	−0.004
	(0.003)	(0.003)
Distance: k-Means	0.011	−0.003
	(0.003)	(0.003)
Distance: K-S	−0.011	−0.008
	(0.003)	(0.003)
Distance: Reconstruction Loss	0.016	−0.005
	(0.003)	(0.004)
Observations	5,700	5,700

Table 3

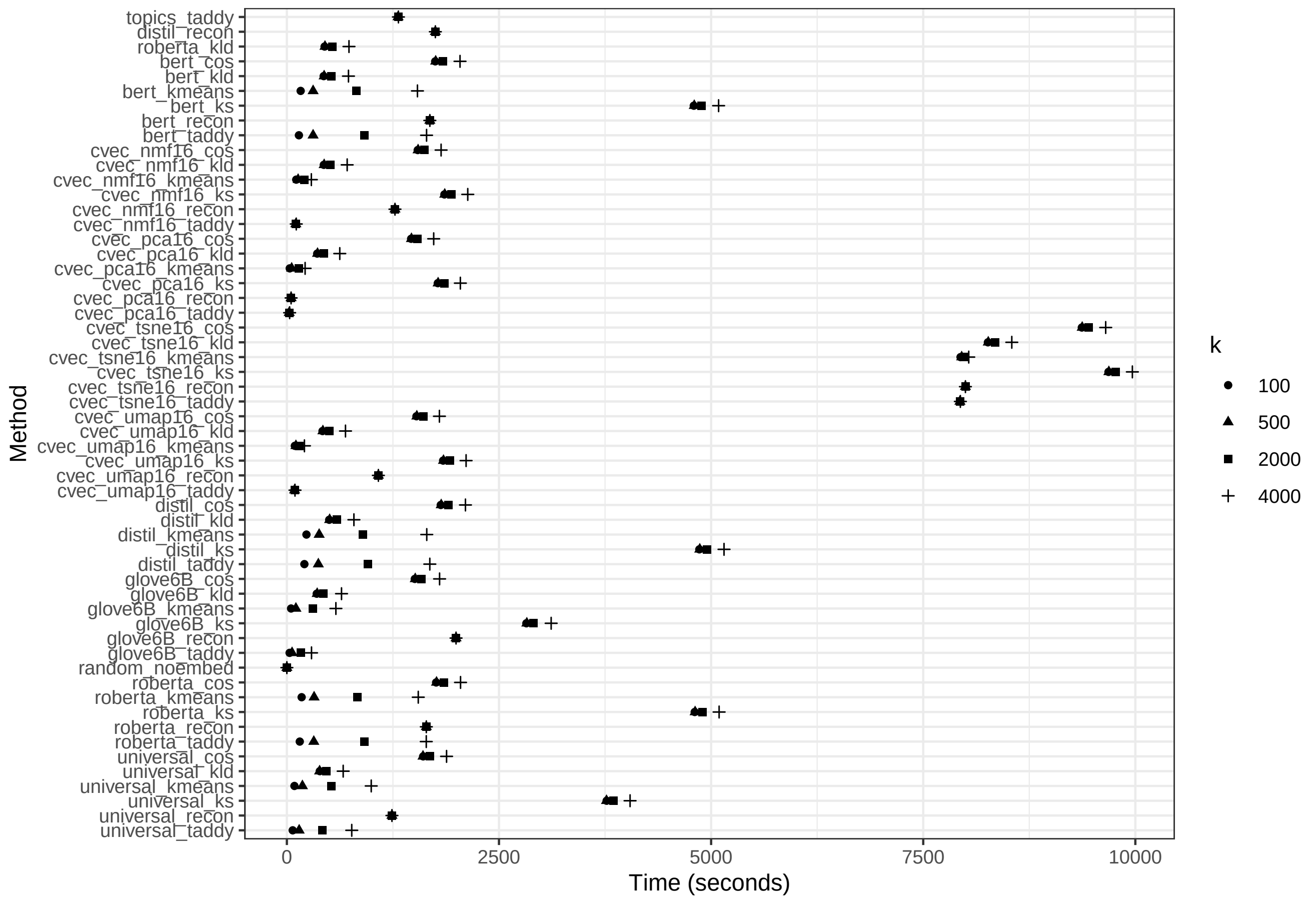

Figure 4: For training sets of 100, 500, 2000, and 4000 observations, the 56 methods (other than random sampling) vary widely in how long they require to select observations. Taddy (2013) performs near the median.

3.2 Computation Time

To the final question, we perform a time benchmarking analysis of our Executive Orders application and show that there is quite large variation in how long a given method takes to produce a training set. Taddy (2013) method performs close to the median. It is around the 53rd percentile for 100 samples and around the 47th percentile for 4000 samples. The method combining DistilBERT and minimizing reconstruction error is about 33% slower than Taddy (2013), though it performs much better. On the other hand, the method combining RoBERTa and minimizing KL Divergence, which similarly outperforms Taddy, requires only 55% as much time.

3.3 Small Absolute Gains in Accuracy are Meaningful

We acknowledge that an increase in accuracy from 0.73 to 0.75, as in our StockTwits application, seems small. But in a data set of 10,000 documents, that is an additional 200 documents correctly classified. Many state-of-the-art models in computer science produce accuracy gains of a handful of thousandths, and the fact that we can produce these gains at no additional cost to the researcher we believe is significant.

That increase in 0.02 may also represent a bigger increase than the number suggest. Many tasks in the social sciences may have low upper-bounds of potential accuracy – no model will ever be extremely accurate at identifying something as complex as partisanship. If the range of possible AUC is 0.5 to 0.1, an increase in 0.02 is 4% of the total range; if political science tasks are more challenging such that the range of possible AUC is 0.5 to 0.8, then an increase of 0.02 is 6.7% of the total range, which is even more significant.

3.4 Beyond Accuracy: Evaluating Model Outputs

In our simulations and results we focus on model accuracy, specifically the Area Under the Receiver-Operator Characteristic (AUC), as our evaluative target. AUC is one of the standard measures of accuracy in the machine learning community, but in political science we often want more out of our machine learning outputs. Since we typically use these predictions as variables in a regression model, often aimed at obtaining causal inference, we may also care about targets like calibration and homoskedasticity. For an extended discussion of these complementary targets in a substantive applied example, see (Kaufman and Rogowski, Forthcoming).

References

Cer, D., Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Céspedes, S. Yuan, C. Tar, et al. 2018. "Universal sentence encoder." *arXiv preprint arXiv:1803.11175.*

Daszykowski, M., B. Walczak, and D. Massart. 2002. "Representative subset selection." *Analytica chimica acta* 468 (1): 91–103.

Han, K., Y. Wang, C. Zhang, C. Li, and C. Xu. 2018. "Autoencoder inspired unsupervised feature selection." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 2941–2945. IEEE.

Kaufman, A. R., and J. C. Rogowski. Forthcoming. "Divided Government, Strategic Substitution, and Presidential Unilateralism." *American Journal of Political Science.*

Maaten, L. Van der, and G. Hinton. 2008. "Visualizing data using t-SNE." *Journal of machine learning research* 9 (11).

Pan, F., W. Wang, A. K. Tung, and J. Yang. 2005. "Finding representative set from massive data." In *Fifth IEEE International Conference on Data Mining (ICDM'05),* 8–pp. IEEE.

Panda, R., A. Das, and A. K. Roy-Chowdhury. 2016. "Embedded sparse coding for summarizing multi-view videos." In *2016 IEEE international conference on image processing (ICIP),* 191–195. IEEE.

Reimers, N., and I. Gurevych. 2019. "Sentence-bert: Sentence embeddings using siamese bert-networks." *arXiv preprint arXiv:1908.10084.*

Taddy, M. 2013. “Measuring political sentiment on Twitter: Factor optimal design for multinomial inverse regression.” *Technometrics* 55 (4): 415–425.

———. 2012. “On estimation and selection for topic models.” In *Artificial Intelligence and Statistics,* 1184–1193. PMLR.