Improving Probabilistic Models in Text Classification via Active Learning*

Mitchell Bosley[†‡] Saki Kuzushima[†§] Ted Enamorado[¶]
Yuki Shiraito[‖]

First draft: September 10, 2020
This draft: September 23, 2022

Abstract

Social scientists often classify text documents to use the resulting labels as an outcome or a predictor in empirical research. Automated text classification has become a standard tool, since it requires less human coding. However, scholars still need many human-labeled documents to train automated classifiers. To reduce labeling costs, we propose a new algorithm for text classification that combines a probabilistic model with active learning. The probabilistic model uses both labeled and unlabeled data, and active learning concentrates labeling efforts on difficult documents to classify. Our validation study shows that the classification performance of our algorithm is comparable to state-of-the-art methods at a fraction of the computational cost. Moreover, we replicate two recently published articles and reach the same substantive conclusions with only a small proportion of the original labeled data used in those studies. We provide *activeText*, an open-source software to implement our method.

*We thank Ken Benoit, Yaoyao Dai, Chris Fariss, Yusaku Horiuchi, Kosuke Imai, Walter Mebane, Daichi Mochihashi, Kevin Quinn, and audiences at the 2020 Annual Meeting of the American Political Science Association, the 2021 Annual Meeting of the Midwest Political Science Association, the 11th Annual Conference on New Directions in Analyzing Text as Data, and the 2022 Summer Meeting of the Japanese Society for Quantitative Political Science, and seminar participants at the University of Michigan and members of the Junior Faculty Workshop at Washington University in St. Louis for useful comments and suggestions.

[†]These authors have contributed equally to this work.

[‡]Ph.D. Candidate, Department of Political Science, University of Michigan. Email: mcbosley@umich.edu.

[§]Ph.D. Candidate, Department of Political Science, University of Michigan. Email: skuzushi@umich.edu

[¶]Assistant Professor, Department of Political Science, Washington University in St. Louis. Siegle Hall, 244. One Brookings Dr. St Louis, MO 63130-4899. Phone: 314-935-5810, Email: ted@wustl.edu, URL: www.tedenamorado.com.

[‖]Assistant Professor, Department of Political Science, University of Michigan. Center for Political Studies, 4259 Institute for Social Research, 426 Thompson Street, Ann Arbor, MI 48104-2321. Phone: 734-615-5165, Email: shiraito@umich.edu, URL: shiraito.github.io.

Introduction

As the amount and diversity of available information have rapidly increased, social scientists are increasingly resorting to multiple forms of data to answer substantive questions. In particular, the use of text-as-data in social science research has exploded over the past decade.[1] Document classification has been the primary task in political science, with researchers classifying documents such as legislative speeches (Peterson and Spirling, 2018; Motolinia, 2021), correspondences to administrative agencies (Lowande, 2018, 2019), public statements of politicians (Airoldi et al., 2007; Stewart and Zhukov, 2009), news articles (Boydstun, 2013), election manifestos (Catalinac, 2016), social media posts (King et al., 2017), treaties (Spirling, 2012), religious speeches (Nielsen, 2017), and human rights text (Cordell et al., 2021; Greene et al., 2019) into two or more categories. Researchers use the category labels of documents produced by the classification task as the outcome or predictive variable to test substantive hypotheses.

Statistical methods are used for document classification. Although text data in political science is typically smaller than data in some other fields (where millions of documents are common), the cost of having human coders categorize all documents is still prohibitively high. Relying on automated text classification allows researchers to avoid classifying all documents in their data set manually.

Broadly speaking, there are two types of classification methods: supervised and unsupervised algorithms. Supervised approaches use labels from a set of hand-coded documents to categorize unlabeled documents, whereas unsupervised methods cluster documents without needing labeled documents. Both of these methods have downsides, however: in the former, hand-coding documents is labor-intensive and costly; in the latter, the substantive interpretation of the categories discovered by the clustering process can be difficult.

Supervised methods are more popular in political science research because substantive interpretability is important in using category labels to test substantive hypotheses, and justifies the cost associated with labeling many documents manually. For example, Gohdes (2020) hand-labeled about 2000 documents, and Park et al. (2020) used 4000 human-coded documents. These numbers are much smaller than the size of their entire data sets ($65,274$ and $2,473,874$, respectively), however, having human coders label thousands of (potentially long and complicated) documents still requires a large amount of researchers' time and effort.

We propose *activeText*, a new algorithm that augments a probabilistic mixture model with active learning. We use the mixture model of Nigam et al. (2000) to combine the information from both labeled and unlabeled documents, making use of all available information. In the

[1] See e.g., Grimmer and Stewart (2013) for an excellent overview of these methods in political science.

model, latent classes are observed as labels for labeled documents and estimated as a latent variable for unlabeled documents. Active learning is a technique that reduces the cost of hand-coding. It uses measures of label uncertainty to iteratively flag highly informative documents to reduce the number of labeled documents needed to train an accurate classifier, particularly when the classification categories are imbalanced.

Our validation study shows that our model outperforms Support Vector Machines (SVM), a popular supervised learning model when both models are using active learning. We also show that our algorithm performs favorably in terms of classification accuracy when compared to an off-the-shelf version of Bidirectional Encoder Representations from Transformers (BERT), a state-of-the-art classification model in natural language processing, using several orders of magnitude less computational resources. Furthermore, because our model is generative, it is straightforward to use a researcher's domain expertise, such as keywords associated with a category, to improve text classification.

We also use *activeText* to replicate two published political science studies and show that the authors of these papers could have reached the same substantive conclusions with fewer labeled documents. The first study is Gohdes (2020), which focuses on the relationship between internet access and the form of state violence. The second study is Park et al. (2020), which analyzes the association (or the lack thereof) between information communication technologies (ICTs) and the U.S. Department of State's reports on human rights. For both studies, we replicate their text classification tasks using *activeText* and conduct the same empirical analyses using the document labels. Our replication analysis recovers their original conclusions—a higher level of internet access is associated with a larger proportion of targeted killings, and ICTs are not associated with the sentiment of the State Department's human rights reports, respectively—using far fewer labeled documents. These replication exercises demonstrate that *activeText* performs well on complex documents commonly used in political science research, such as human rights reports.

We provide an **R** package called *activeText* with the goal of providing researchers from all backgrounds with easily accessible tools to minimize the amount of hand-coding of documents and improve the performance of classification models for their own work.

Before proceeding to a description of our algorithm and analysis, we first offer an accessible primer on the use of automated text classification. We introduce readers to several basic concepts in machine learning: tokenization, preprocessing, and the encoding of a corpus of text data into a matrix; the difference between supervised and unsupervised learning, between discriminative and generative models, and between active and passive learning; and a set of tools for the evaluation of classification models. Readers who are already well acquainted with these concepts may prefer to skip directly to the description of our model in

Section The Method.

Using Machine Learning for Text Classification

Encoding Text in Matrix Form

Suppose that a researcher has a collection of social media text data, called a corpus, and wishes to classify whether each text in a corpus is political (e.g., refers to political protest, human rights violations, unfavorable views of a given candidate, targeted political repression, etc.) or not solely based on the words used in a given observation.[2] Critically, the researcher does not yet know which of the texts are political or not at this point.

The researcher must first choose how to represent text as a series of *tokens*, and decide which tokens to include in their analysis. This involves a series of sub-choices, such as whether each token represents an individual word (such as "political") or a combination of words (such as "political party"), whether words should be stemmed or not (e.g., reducing both "political" and "politics" their common stem "politic"), and whether to remove stop-words (such as "in", "and", "on", etc.) that are collectively referred to as *pre-processing.*[3]

The researcher must then choose how to encode information about these tokens in matrix form. The most straightforward way to accomplish this is using a *bag-of-words* approach, where the corpus is transformed into a document-feature matrix (DFM) $\mathbf{X}$ with n rows and m columns, where n is the number of documents and m is the number of tokens, which are more generally referred to as features.[4] Each element of the DFM encodes the frequency that a token occurs in a given document.[5] Once the researcher chooses how to encode their corpus as a matrix, she is left with a set of features corresponding to each document $\mathbf{X}$ and an unknown vector of true labels Y, where each element of Y indicates whether a given document is political or not. Then, we can repose the classification question as follows: given

[2]For simplicity, the exposition here focuses on a binary classification task, however, our proposed method can be extended to multiple classes e.g., classifying a document as either a positive, negative, or neutral position about a candidate. See Sections The Method and Reanalysis with Fewer Human Annotations, and Supplementary Information (SI) C for more details.

[3]For a survey of pre-processing techniques and their implications for political science research, see Denny and Spirling (2018).

[4]Note that in the machine learning literature, the concept typically described by the term "variable" is communicated using the term "feature."

[5]An alternative to the bag-of-words approach is to encode tokens as *word embeddings*, where in addition to the matrix summarizing the incidences of words in each document, neural network models are used to create vector representations of each token. In this framework, each token is represented by a vector of some arbitrary length, and tokens that are used in similar contexts in the corpus (such as "minister" and "cabinet") will have similar vectors. While this approach is more complicated, it yields considerably more information about the use of words in the corpus than the simple count that the bag-of-words approach does. For an accessible introduction to the construction and use of word embeddings in political science research, see Rodriguez and Spirling (2022). For a more technical treatment, see Pennington et al. (2014).

$\mathbf{X}$, how might we best learn Y, that is, whether each document is political or not?

Supervised vs. Unsupervised Learning

A researcher must then choose whether to use a supervised or unsupervised approach to machine learning.[6] The supervised approach to this problem would be to (1) obtain true labels of some of the documents using human coding e.g., an expert classifies documents such as the following news headline by CNN: "White House says Covid-19 policy unchanged despite President Biden's comments that the 'pandemic is over"' as political or not; (2) learn the relationship between the text features encoded in the matrix $\mathbf{X}$ and the true label encoded in the vector Y for the documents with known labels. In other words, it learns the importance of words such as "policy", "President", "Biden", "pandemic" in explaining whether a document refers to politics or not;[7] and (3) using the learned association between the text data and the known labels, predict whether the remaining documents in the corpus (that is, those that were not coded by a human) are political or not.

In contrast, an unsupervised approach would *not* obtain the true labels of some of the documents. Rather, a researcher using an unsupervised approach would choose a model that *clusters* documents from the corpus that have common patterns of word frequency.[8] Using the assignment of documents to clusters, the researcher would then use some scheme to decide which of the clusters corresponds to the actual outcome of interest: whether a document is political or not.

The main advantage of a supervised approach over an unsupervised approach is the direct interpretability of results, since it requires the translation of clusters to classifications. This also allows for a more straightforward evaluation of model performance in terms of the distance between the predictions made by the supervised learning algorithm and the true values of Y. Because such an objective measure does not exist in unsupervised learning, the researcher needs to rely on heuristics to assess the adequacy of the algorithm (Hastie et al., 2009).[9]

On the other hand, the main disadvantage of a supervised approach is that obtaining labels for the documents in the corpus is often time-consuming and costly. For example, it requires expert knowledge to classify each document to be either political or non-political.

[6] For a comprehensive discussion on supervised and unsupervised algorithms for the analysis of text as data, we refer the interested reader to Grimmer et al. (2022).

[7] That is, learn $P(Y_{\text{labeled}}|\mathbf{X}_{\text{labeled}})$. This can be accomplished with a variety of models, including e.g. linear or logistic regression, support vector machines (SVM), Naive Bayes, K-nearest neighbor, etc.

[8] Examples of clustering algorithms include K-means and Latent Dirichlet Allocation (LDA).

[9] In most political science applications of unsupervised learning techniques, the author either is conducting an exploratory analysis and is therefore uninterested in classification, or performs an *ad hoc* interpretation of the clusters by reading top examples of a given cluster, and on that basis infers the classification from the clustering (Knox et al., 2022).

Researchers using an unsupervised approach instead will avoid this cost since they do not require a set of labels *a priori*.

Semi-supervised methods combine the strengths of supervised and unsupervised approaches to improve classification (Miller and Uyar, 1996; Nigam et al., 2000). These methods are particularly useful in situations where there is a large amount of unlabeled data, and acquiring labels is costly. A semi-supervised model proceeds similarly to the supervised approach, with the difference being that the model learns the relationship between the matrix of text data $\mathbf{X}$ and the classification outcome Y using information from both the labeled and unlabeled data.[10] Since a supervised approach learns the relationship between the labels and the data solely based on the labeled documents, a classifier trained with a supervised approach maybe less accurate than if it were provided information from both the labeled and unlabeled documents (Nigam et al., 2000).

Discriminative vs. Generative Models

In addition to choosing a supervised, unsupervised, or semi-supervised approach, a researcher must also choose whether to use a discriminative or generative model. As noted by Ng and Jordan (2001) and Bishop and Lassarre (2007), when using a discriminative model (e.g., logistic regression, SVM, etc.), the goal is to directly estimate the probability of the classification outcomes Y given the text data $\mathbf{X}$ i.e., directly estimate $p(Y|\mathbf{X})$. In contrast, when using a generative model (e.g., Naive Bayes), learning the relationship between the Y and $\mathbf{X}$ is a two-step process. In the first step, the likelihood of the matrix of text data $\mathbf{X}$ and outcome labels Y is estimated given the data and a set of parameters θ that indicate structural assumptions about how the data is generated i.e., $p(\mathbf{X}, Y|\theta)$ is directly estimated. In the second step, the researcher uses Bayes' rule to calculate the probability of the outcome vector given the features and the learned distribution of the parameters i.e., $p(Y|\mathbf{X};\theta)$.

In addition to allowing for the use of unlabeled data (which reduces labeling costs), one of the main benefits of a generative rather than a discriminative model is that the researcher can include information they know about the data generating process by choosing appropriate functional forms.[11] This can help prevent overfitting when the amount of data in a corpus is small.[12] Conversely, because it is not necessary to model the data generating process directly,

[10] While Y is not observed for the unlabeled data, these observations do contain information about the joint distribution of the features $\mathbf{X}$, and as such can be used with labeled data to increase the accuracy of a text classifier (Nigam et al., 2000).

[11] This is particularly true when e.g., the researcher knows that the data has a complicated hierarchical structure since the hierarchy can be incorporated directly into the generative model.

[12] Overfitting occurs when a model learns to predict classification outcomes based on patterns in the training set (i.e., the data used to fit the model) that does not generalize to the broader universe of cases to be classified. A model that is overfitted may predict the correct class with an extremely high degree of accuracy for items in the training set, but will perform poorly when used to predict the class for items that

the main benefit of a discriminative rather than generative model is simplicity (in general it involves estimating fewer parameters). Discriminative models are therefore appropriate in situations where the amount of data in a corpus is very large, and/or when the researcher is unsure about the data-generating process, which could lead to mis-specification (Bishop and Lassarre, 2007).[13]

Model Evaluation

A researcher must also decide when she is satisfied with the predictions generated by the model. In most circumstances, the best way to evaluate the performance of a classification algorithm is to reserve a subset of the corpus for validation, which is sometimes referred to as validation and/or test set. At the very beginning of the classification process, a researcher puts aside and label a set of randomly chosen documents that the active learning algorithm does not have access to.[14] Then, after training the model on the remainder of the documents (often called the training set), the researcher should generate predictions for the documents in the validation set using the trained model. By comparing the predicted labels generated by the model to the actual labels, the researcher can evaluate how well the model does at predicting the correct labels.

A common tool for comparing the predicted labels to the actual labels is a *confusion matrix*. In a binary classification setting, a confusion matrix will be a 2 by 2 matrix, with rows corresponding to the actual label, and the columns corresponding to the predicted label. Returning to our running example, imagine that the classification is to predict whether documents are political or not, Table 1 shows the corresponding confusion matrix. In this scenario, True Positives (TP) are the number of documents that the model predicts to be about politics and that is in fact labeled as such. Correspondingly, True Negatives (TN), are the number of documents that the model predicts to be non-political and is labeled as such in the validation set. A False Negative (FN) occurs when the model classifies a document as non-political, but according to the validation set, the document is about politics. Similarly, a False Positive (FP) occurs when the model classifies as political a document that is non-political.

Using the confusion matrix, the researcher can calculate a variety of evaluation statistics. Some of the most common of these are accuracy, precision, and recall. Accuracy is the

the model has not seen before.

[13]Another benefit of generative models is that they can yield better estimates of how certain we are about the relationship between the outcome and the features. This is the case when a researcher uses an inference algorithm like Markov Chain Monte Carlo (MCMC) that learns the entire distribution for each of the parameters, rather than only point estimates.

[14]It is important to use a set-aside validation set for testing model performance, rather than a subset of the documents used to train the model, to avoid *overfitting*.

		Predicted Label	
		Political	Non-political
Actual Label	Political	True Positive (TP)	False Negative (FN)
	Non-political	False Positive (FP)	True Negative (TN)

Table 1: **Confusion Matrix: Comparison of the Predictions of a Classifier to Documents' True Labels**

proportion of documents that have been correctly classified. Precision is used to evaluate the false positivity rate and is the proportion of the model's positive classifications that are true positives. As the number of false positives increases (decreases), precision decreases (increases). Recall is used to evaluate the false negativity rate, and is the proportion of the actual positive documents that are true positives. As the number of false negatives increases, recall decreases, and *vice-versa*. Accuracy, precision, and recall can be formally calculated as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \qquad \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

When the proportion of political and non-political documents in a corpus is balanced, accuracy is an adequate measure of model performance. However, it is often the case in text classification that the corpus is unbalanced, and the proportion of documents associated with one class is low. When this is the case, accuracy does a poor job at model evaluation. Consider the case when 99 percent of documents are non-political, and 1 percent are about politics. A model which simply predicts that all documents belong to the non-politics class would have an accuracy score of 0.99, but would be poorly suited to the actual classification task. In contrast, the precision and recall rates would be 0, which would signal to the researcher that the model does a poor job at classifying documents as political. Precision and recall are not perfect measures of model performance, however. There is a fundamental trade-off involved in controlling the false positivity and false negativity rates: you can have few false positives if you are content with an extremely high number of false negatives, and you can have few false negatives if you are content with an extremely high number of false positives.

Recognizing this trade-off, researchers often combine precision and recall scores to find a model that has the optimal balance of the two. One common way of combining the two is an F1 score, which is the harmonized mean of precision and recall. Formally, the F1 score is calculated as:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 score evenly weights precision and recall, and so a high F1 score would indicate that

both the false negativity and false positivity rate are low. It is worth noting these evaluation measures (accuracy, precision, recall, and the F1 score) are computed using labeled data ("ground truth"), which in practice, are available only for a limited subset of the records.

Active vs. Passive Learning

Finally, if the researcher in our running example decides to use a supervised or semi-supervised approach for predicting whether documents in their corpus are political or not, the next step is to decide how many documents to label, and how to choose them. Since labeling is the bottleneck of any classification task of this kind, it is critical that she also selects an approach to label observations that minimizes the number of documents to be labeled in order to produce an accurate classifier.

There are two popular strategies on how to retrieve cases to be labeled: 1) passively and 2) actively. The difference between a passive and an active approach amounts to whether the researcher randomly chooses which documents to label (i.e., choose documents *passively*), or whether to use some selection scheme (i.e., choose documents *actively*). Ideally, an active approach must require fewer labels than the number of randomly labeled data sufficient for a passive approach to achieve the same level of accuracy.

Cohn et al. (1994) and Lewis and Gale (1994) established that a good active learning algorithm should be fast, and should reliably choose documents for labeling that provide more information to the model than a randomly chosen document, particularly in situations when the amount of labeled data is scarce.[15] One of the most studied active learning approaches is called *uncertainty sampling* (Lewis and Gale, 1994; Yang et al., 2015), a process where documents are chosen for labeling based on how uncertain the model is about the correct classification category for each document in the corpus.[16]

As noted above, an active learning process using uncertainty sampling alternates between estimating the probability that each document belongs to a particular classification outcome, sampling a subset of the documents that the model is most uncertain about for labeling,[17] then estimating the probabilities again using the information from the newly labeled documents. In our running example, a researcher is interested in classifying documents

[15] See also Dasgupta (2011); Settles (2011); Hanneke (2014); Hino (2021) and the references therein.

[16] This is just one of many possible approaches. Other uncertainty-based approaches to active learning include query-by-committee, variance reduction, expected model change, etc. We refer the interested reader to Settles (2011) for an accessible review on active learning and Hanneke (2014) for a more technical exposition.

[17] While in our presentation, we have focused on instances of labeling one observation per iteration, exactly how many observations to select and label at each active iterations is also an important practical consideration for any researcher. As noted by Hoi et al. (2006), to reduce the cost of retraining the model per instance of labeling, labeling many documents per iteration (as a batch) is the best approach. This is especially important when working with a large amount of data.

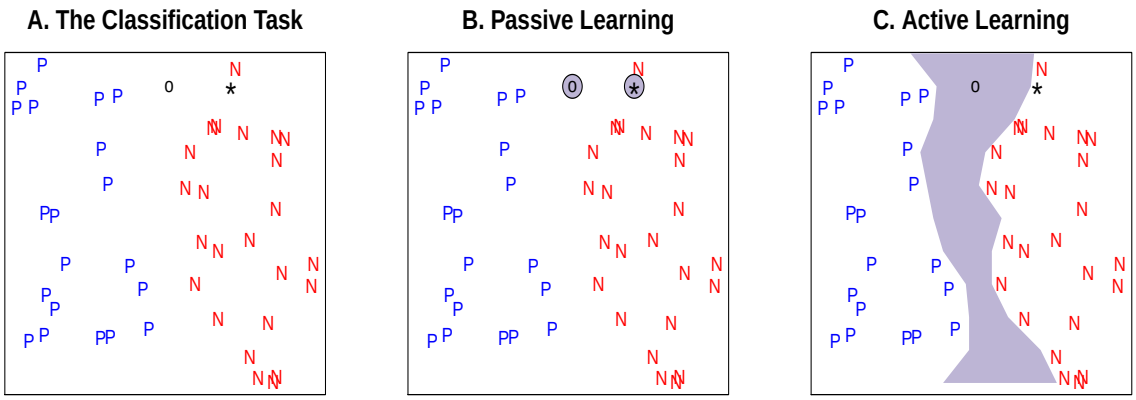

Figure 1: **Passive vs Active Learning.** For a classifier defined in two dimensions, Panel A illustrates the task: classify unlabeled documents (denoted by ∘ and ∗) as Political (P) or Non-political (N). A a passive learning algorithm will request the labels of ∘ and ∗ with equal probability (Panel C). In contrast, in active learning approach, ∘ will be prioritized for labeling as it is located in the region where the classifier is most uncertain (shaded region).

as political (P) or non-political (N), and needs to decide how to prioritize her labeling efforts. As shown in Figure 1 (Panel A), imagine there are two new data points to be labeled (denoted by "∘" and "∗"). A passive learning algorithm would give equal labeling priority to both (Panel B). However, an active approach would give priority to "∘" as the classifier is most uncertain about the label of "∘" if compared to "∗" (which is surrounded by many non-political documents).

A critical question for a researcher using an iterative algorithm is when to stop labelling. Many active learning algorithms resort to heuristics such as a fixed-budget approach, which stops when the number of newly labeled data points reaches a predetermined size. The problem with such an approach is that it may lead to under- or over-sampling.[18] One popular strategy is to randomly label a subset of documents at the beginning of the process, which is then used for assessing the performance of the classifier on data that the model has not seen.[19] With this approach, the process stops when the difference in measures of out-of-sample accuracy between two consecutive iterations does not surpass a certain threshold pre-established by the researcher (e.g., the F1 score does not improve in more than 0.01 units from iteration to iteration) (Altschuler and Bloodgood, 2019). If labeled data does not exist or cannot be set aside for testing due to its scarcity, a stopping rule where the algorithm stops once in-sample predictions generated by the model (i.e., using the documents that have been labeled by the researcher during the active learning process) do not change from one

[18]This is due to the fact the fixed budget has not been set using an optimality criterion other than to stop human coding at some point. See Ishibashi and Hino (2020) for further discussion of this point.

[19]For a discussion of this approach in our own application, see Section Model Evaluation.

iteration to the next. This is often referred to as a stability-based method (Ishibashi and Hino, 2020).

With all these concepts in mind, in the next section we describe our proposed approach with a special focus on its flexibility that it affords a researcher to both balance the tradeoffs of working with labeled and unlabeled data, and use existing domain expertise to improve classification with the use of keyword upweighting.

The Method

In this section, we present our modeling strategy and describe our active learning algorithm. For the probabilistic model (a mixture model for discrete data) at the heart of the algorithm, we build on the work of Nigam et al. (2000), who show that probabilistic classifiers can be augmented by combining the information coming from labeled and unlabeled data. In other words, our model makes the latent classes for the unlabeled data interpretable by connecting them to the hand-coded classes from the labeled data. It also takes advantage of the fact that the unlabeled data provides more information about the features used to predict the classes for each document. As we will discuss below, we insert our model into an active learning algorithm and use the Expectation-Maximization (EM) algorithm (Dempster et al., 1977) to maximize the observed-data log-likelihood function and estimate the model parameters.

Model

Consider the task of classifying N documents as one of two classes (e.g., political vs non-political). Let $\mathbf{D}$ be a $N \times V$ document feature matrix, where V is the size of features. We use $\mathbf{Z}$, a vector of length N, where each entry represents the latent classes assigned to each document. If a document i is assigned to the kth class, we have that $Z_i = k$, where $k \in \{0, 1\}$ e.g., in our running example, $k = 1$ represents the class of documents about politics, and $k = 0$ those that are non-political. Because we use a semi-supervised approach, it can be the case that some documents are already hand-labeled. This means that the value of Z_i is known for the labeled documents and is unknown for unlabeled documents. To facilitate exposition, we assume that the classification goal is binary, however, our approach can be extended to accommodate for 1) multiclass classification setting, where $k > 2$ and each document needs to be classified into one of the k classes e.g., classifying news articles into 3 classes: politics, business, and sports; and 2) modeling more than two classes but keeping the final classification to be binary. In other words, a hierarchy that maps multiple sub-classes into one class e.g., collapsing the classification of documents that are about business and sports into a larger class (non-politics), and letting the remaining documents to be about politics (the main category of interest). (For more details, see SI A, B, and C).

The following sets of equations summarize the model:

Labeled Data

$$
\begin{aligned}
Z_i = k &\sim \text{hand-coded}, \quad k \in \{0, 1\} \\
\eta_{\cdot k} &\overset{i.i.d}{\sim} Dirichlet(\boldsymbol{\beta}_k) \\
\mathbf{D}_{i\cdot} | Z_i = k &\overset{i.i.d}{\sim} Multinomial(n_i, \boldsymbol{\eta}_{\cdot k})
\end{aligned}
$$

$+$

$\lambda \cdot$ Unlabeled Data

$$
\begin{aligned}
\pi &\sim Beta(\alpha_0, \alpha_1) \\
Z_i = k &\overset{i.i.d}{\sim} Bernoulli(\pi), \quad k \in \{0, 1\} \\
\eta_{\cdot k} &\overset{i.i.d}{\sim} Dirichlet(\boldsymbol{\beta}_k) \\
\mathbf{D}_{i\cdot} | Z_i = k &\overset{i.i.d}{\sim} Multinomial(n_i, \boldsymbol{\eta}_{\cdot k})
\end{aligned}
$$

If document i is unlabeled, we first draw $\pi = p(Z_i = 1)$, the overall probability that any given document belongs to the first class (e.g., political documents), from a Beta distribution with hyperparameters α_0 and α_1. Similarly, for the other class (e.g., non-political documents), we have that $1 - \pi = p(Z_i = 0)$. Given π, for each document indexed by i, we draw the latent cluster assignment indicator Z_i from a Bernoulli distribution. Then, we draw features for document i from a multinomial distribution governed by the vector $\boldsymbol{\eta}_{\cdot k}$, where $\eta_{vk} = p(D_{iv} | Z_i = k)$, whose prior is the Dirichlet distribution. If document i is labeled, the main difference with the unlabeled data case is that Z_i has been hand-coded, and as a result, we do not draw it from a Bernoulli distribution but the rest of the model's structure remains the same.

It is worth emphasizing that one of the most notorious problems with the implementation of supervised and semi-supervised approaches is the scarcity of labeled data, especially if compared to the abundance of unlabeled data. Due to this imbalance problem, for any classifier to be able to extract signal from the labeled data and not be informed by unlabeled data alone, it is key to devise ways to increase the relative importance of the labeled data. Otherwise, the unlabeled data will mute the signal coming from the labeled data. Following Nigam et al. (2000), we down-weight information from unlabeled documents by $\lambda \in [0, 1]$. Note that when the λ is equal to 1, the model treats each document equally, regardless of

whether the document is labeled deterministically by a human, or probabilistically by the algorithm. As λ moves from 1 towards 0, the model increasingly down-weights the information that the probabilistically labeled documents contribute to the estimation of $\boldsymbol{\eta}$ and π, such that when λ is 0, the model *ignores* all information from the probabilistically labeled documents and therefore becomes a supervised algorithm (see SI A). Finally, because the observed data log-likelihood of our model is difficult to maximize, we use the EM algorithm to estimate the parameters.[20]

Active Learning

Our active learning algorithm (see Algorithm 1) can be split into the following steps: *estimation* of the probability that each unlabeled document belongs to the positive class, *selection* of the unlabeled documents whose predicted class is most uncertain, and *labeling* of the selected documents by human coders. The algorithm iterates until a stopping criterion is met (Section). We also describe an optional keyword upweighting feature, where a set of user-provided keywords provide prior information about the likelihood that a word is generated by a given class to the model. These keywords can either be provided at the outset of the model or identified during the active learning process.

Estimation

In the first iteration, the model is initialized with a small number of labeled documents.[21] The information from these documents is used to estimate the parameters of the model: the probability of a document being of class 1 (π), and the probability of generating each word given a class, the $V \times 2$ matrix $\boldsymbol{\eta}$. From the second iteration on, we use information from both labeled and unlabeled documents to estimate the parameters using the EM algorithm, with the log-likelihood of unlabeled documents being down-weighted by λ, and with the $\boldsymbol{\eta}$ and π values from the previous iteration as the initial values. Using the estimated parameters, we compute the posterior probability that each unlabeled document belongs to class 1.

Selection

Using the predicted probability that each unlabeled document belongs to class 1, we use Shannon Entropy to determine which of the probabilistically labeled documents that it was least certain about. In the binary classification case, this is the equivalent of calculating the absolute value of the distance of the class 1 probability and 0.50 for each document. Using this criterion, the model ranks all probabilistically labeled documents in descending order of

[20] For a full derivation of the EM algorithm, see SI A.

[21] While we assume that these documents are selected randomly, the researcher may choose any subset of labeled documents with which to initialize the model.

Algorithm 1: Active learning with EM algorithm to classify text

Result: Obtain predicted classes of all documents.
Randomly select a small subset of documents, and ask humans to label them;
[**Active Keyword**]: Ask humans to provide initial keywords;
while *Stopping conditions are not met yet* **do**
  (1) [**Active Keyword**]: Up-weight the important of keywords associated with a class;
  (2) Predict labels for unlabeled documents using EM algorithm;
  (3) Select documents with the highest uncertainty among unlabeled documents, and ask humans to label them;
  (4) [**Active Keyword**]: Select words most strongly associated with each class, and ask humans to label them;
  (5) Update sets of labeled and unlabeled documents for the next iteration;
end

uncertainty. The n most uncertain documents are then selected for human labeling, where n is the number of documents to be labeled by humans at each iteration.

Labeling

A human coder reads each document selected by the algorithm and imputes the "correct" label. For example, the researcher may be asked to label as political or non-political each of the following sentences:

> The 2020 Presidential Election had the highest turnout in US history.
> Qatar is ready to host the FIFA World Cup this coming November.

These newly-labeled documents are then added to the set of human-labeled documents, and the process is repeated from the estimation stage.

Stopping Rule

Our method is highly modular and supports a variety of stopping rules. This includes an internal stability criterion, where stoppage is based on small amounts of change of the internal model parameters, as well as the use of a small held-out validation set to assess the marginal benefit of labeling additional documents on measures of model evaluation such as accuracy or F1. With either rule, the researcher specifies some bound such that if the change in model parameters or out-of-sample performance is less than the pre-specified bound, then the labeling process ends. We use the out-of-sample validation stopping rule with a bound of 0.01 for the F1 score in our reanalyses in Section Reanalysis with Fewer Human Annotations.

Active Keyword Upweighting

The researcher also has the option to use an active keyword upweighting scheme, where a set of keywords is used to provide additional information. This is done by incrementing elements of the $\boldsymbol{\beta}$ (the prior of $\boldsymbol{\eta}$) by γ, a scalar value chosen by the researcher. In other words, we impose a tight prior on the probability that a given keyword is associated with each class.[22] To build the set of keywords for each class, 1) *activeText* proposes a set of candidate words, 2) the researcher decides whether they are indeed keywords or not,[23] and 3) *activeText* updates the parameters based on the set of keywords.

To select a set of candidate keywords, *activeText* calculates the ratio that each word was generated by a particular class using the $\boldsymbol{\eta}$ parameter. Specifically, it computes $\eta_{vk}/\eta_{vk'}$ for $k = \{0, 1\}$ with k' the opposite class of k, and chooses top m words whose $\eta_{vk}/\eta_{vk'}$ are the highest as candidate keywords to be queried for class k.[24] Intuitively, words closely associated with the classification classes are proposed as candidate keywords. For example, words such as "vote," "election," and "president," are likely to be proposed as the keywords for the political class of documents in the classification between political vs. non-political documents.

After *activeText* proposes candidate keywords, the researcher decides whether they are indeed keywords or not. This is where the researcher can use her expertise to provide additional information. For example, she can decide names of legislators and acronyms of bills as keywords for the political class.[25]

Using the set of keywords for each class, *activeText* creates a $V \times 2$ keyword matrix $\boldsymbol{\kappa}$ where each element $\kappa_{v,k}$ takes the value of γ if word v is a keyword for class k, otherwise 0. Before we estimate parameters in each active iteration, we perform a matrix sum $\boldsymbol{\beta} \leftarrow \boldsymbol{\kappa} + \boldsymbol{\beta}$ to incorporate information from keywords. The keyword approach therefore effectively upweights our model with prior information about words that the researcher thinks are likely to be associated with one class rather than another.

Validation Performance

This section shows the performance comparisons between *activeText*and other classification methods. First, we show comparisons between active vs. passive learning as well as semi-supervised learning vs. supervised learning. For semi-supervised learning, we use *activeText*with $\lambda = 0.001$. For supervised learning, we use active Support Vector Machines

[22]See Eshima et al. (2020) for a similar approach for topic models.

[23]The researcher may also provide an initial set of keywords, and then iteratively adds new keywords.

[24]Words are excluded from candidate keywords if they are already in the set of keywords, or if they are already decided as non-keywords. Thus, no words are proposed twice as candidate keywords.)

[25]See SI H.1 for more discussion about what if the researcher mislabels keywords.

(SVM) from Miller et al. (2020) with margin sampling. Then, we compare classification and time performance between *activeText* and an off-the-shelf version of BERT, a state-of-the-art text classification model. Furthermore, we show how keyword upweighting can improve classification accuracy.

We compare the classification performance on the following documents: internal forum conversations of Wikipedia editors (class of interest: toxic comment), BBC News articles (political topic), the United States Supreme Court decisions (criminal procedure), and Human Rights allegations (physical integrity rights allegation).[26] We use 80% of each dataset for the training data and hold out the remaining 20% for evaluation. Documents to be labeled are sampled only from the training set, and documents in the test set are not included to train the classifier, even in our semi-supervised approach. The out-of-sample F1 score is calculated using the held-out testing data.

Comparison between *activeText* and Active SVM

Figure 2 shows the results from four model specifications, each representing one of the combinations of active or passive learning, and semi-supervised or supervised learning. The first choice is between active learning (solid lines) vs passive learning (dashed lines). In the active sampling, we select the next set of documents to be labeled based on the entropy of the predicted probabilities of the classes when we use our mixture model, and they are selected based on the margin sampling when we use SVM as the underlying classification method. The second choice is between our semi-supervised learning (darker lines) vs. off-the-shelf supervised learning (lighter lines). For the supervised learning, we replicate the results from Miller et al. (2020) which uses SVM as the classifier. Each panel represents model performance in one of four datasets, with the number in parentheses indicating the proportion of documents associated with the class of interest using ground-truth labels in each dataset. The y-axis indicates the average out-of-sample F1 score across 50 Monte Carlo iterations, and the x-axis shows the total number of documents labeled, with 20 documents labeled at each sampling step.[27]

Among the four models, the combination of active learning with the mixture model (*activeText* in Figure 2) performs the best with most of the specifications. The gain from active learning tends to be higher when the proportion of documents in the class of interest is small. On the Wikipedia corpus with the proportion of the positive labels being 9%, active learning outperforms passive learning, particularly when the number of documents labeled is

[26] More information about preprocessing and descriptions about the dataset are in SI D

[27] While we simulate human coders who label all documents correctly at the labeling stage, this may not be the case because humans can make mistakes in practice. SI H.2 shows that honest (random) mistakes in the labeling of documents can hurt the classification performance.

smaller. In SI F, we further examine how the class-imbalance influences the benefit of active learning, by varying the proportion of the positive class between 5% and 50%.[28] It shows that active learning performs better than passive learning consistently when the proportion of one class is 5%.One limitation is that *activeText* did not perform better than SVM on the human rights corpus when the number of documents labeled is small (less than 200 in Figure 2). We examine how the optional keyword labeling can assist such a situation in Benefits of Keyword Upweighting.

Comparison between *activeText* and BERT

In Figure 3, we compare both classification performance and computational time for *activeText*, Active SVM, and BERT, a state-of-the-art text classification model.[29] We trained two sets of models for the F1 and time comparisons, respectively. The left-hand column of panels shows F1 (the y-axis) as a function of the number of documents labeled (the x-axis), as with the results shown in Figure 2. We trained models using 50 random initializations for the *activeText* and Active SVM models. We trained the BERT models using 10 random initializations using V100 GPUs on a cluster computing platform.

The F1 comparison in the left-hand column of Figure 3 shows that for all four of our corpora, *activeText* performs favorably in comparison to our off-the-shelf implementation of the BERT language model. We show that with each of the BBC, Supreme Court, and Wikipedia corpora (the first, third, and fourth rows of panels), we significantly outperform BERT when there are very few documents labeled. As the number of labeled documents increases, BERT as expected performs well and even exceeds the F1 score of *activeText* in the case of Wikipedia. And as shown in the results for the Human Rights corpus (the second row of panels), BERT does outperform *activeText* at all levels of documents labeled.

The right-hand column of panels in Figure 3 shows computational time, rather than F1, as a function of documents labeled. For this analysis, our goal was to compare how long it would take a researcher without access to a cluster computing platform or a high-powered GPU to train these models. To this end, we re-trained the *activeText*, Active SVM, and BERT models on a base model M1 Macbook Air with 8 GB of RAM and 7 GPU cores. While the Active SVM and *activeText* models were trained using a single CPU, we used the recent implementation of support for the GPU in M1 Macs in PyTorch[30] to parallelize the training of the BERT model using the M1 Mac's GPU cores.[31] We also computed the

[28] See SI D for how we generate data with class-imbalance.

[29] For a technical overview of BERT, and the Transformers technology underpinning it, see Devlin et al. (2018) and Vaswani et al. (2017), respectively.

[30] See https://pytorch.org/blog/introducing-accelerated-pytorch-training-on-mac/.

[31] Specifically, we trained a *DistilBERT* model (see Sanh et al. (2019)) for three epochs (the number of passes of the entire training dataset BERT has completed) using the default configuration from the

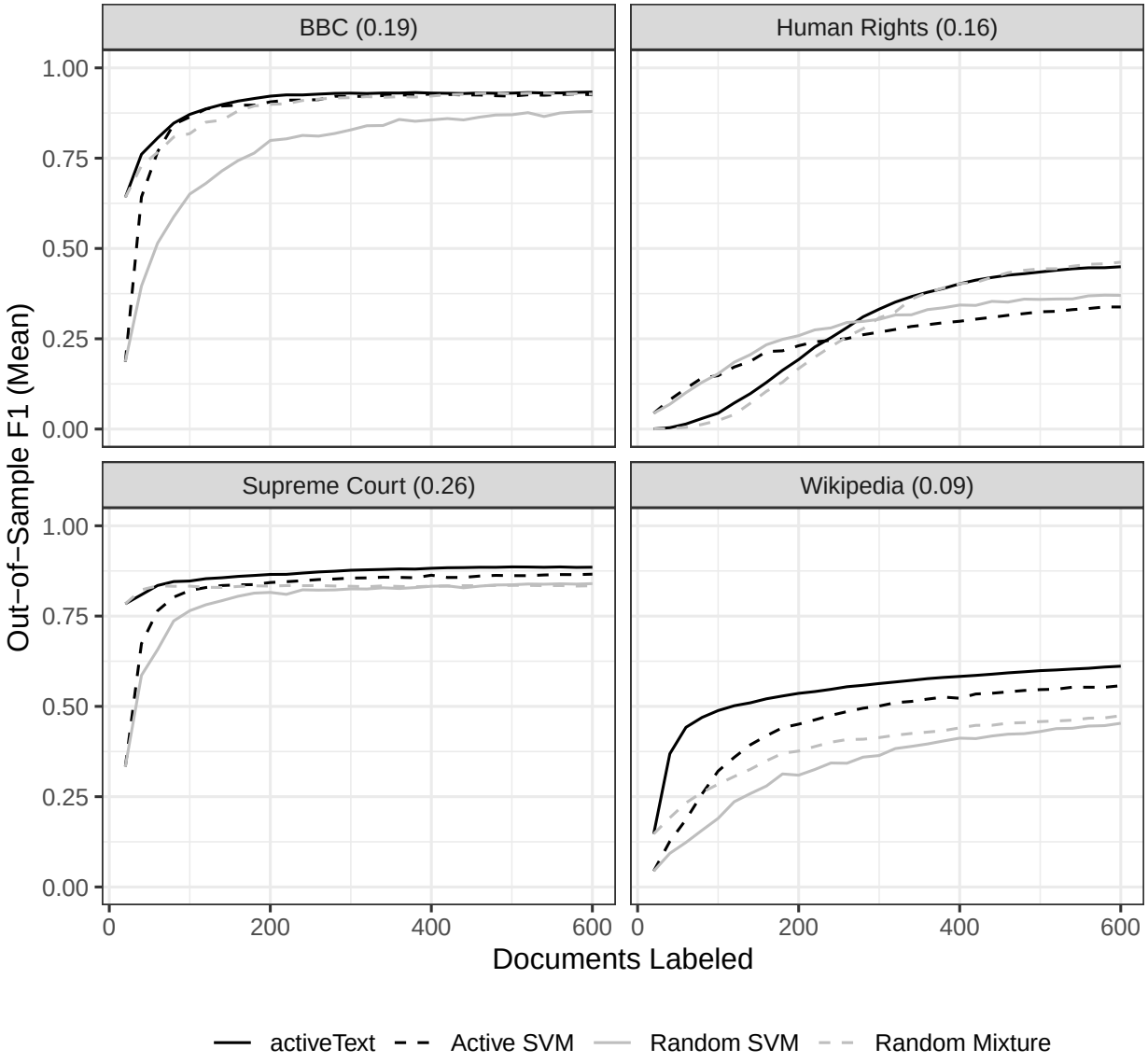

Figure 2: **Comparison of Classification Results across Random and Active Versions of *activeText* and SVM**

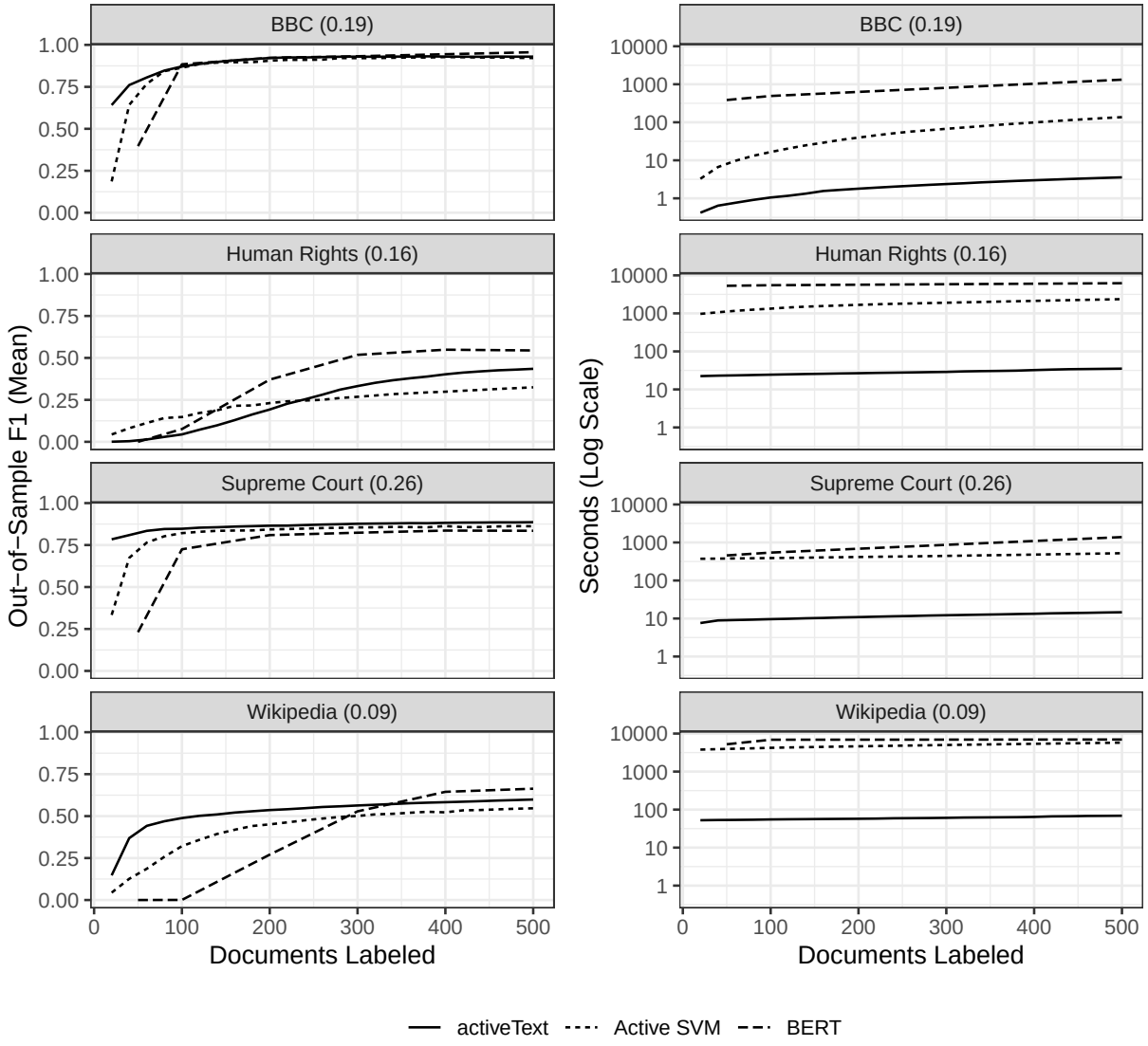

Figure 3: **Comparison of Classification and Time Results across *activeText*, Active SVM, and BERT**

time values *cumulatively* for the *activeText* and Active SVM models, since it is expected that model will be fit over and over again as part of the active learning process, whereas for a model like BERT we expect that the model would only be run once, and as such do not calculate its run-time cumulatively. For the Human Rights and Wikipedia corpora, which each have several hundred thousand entries, we used a random subsample of 50,000 documents. For the Supreme Court and BBC corpora, we used the full samples. Finally, we present the time results in logarithmic scale to improve visual interpretation.

The right-hand panel of Figure 3 shows that the slight advantages of the BERT models come at a cost of several orders of magnitude of computation time. Using the Wikipedia corpus as an example, at 500 documents labeled the baseline *activeText* would have run to convergence 25 times, and the sum total of that computation time would have amounted to just under 100 seconds. With BERT, however, training a model with 500 documents and labeling the remaining 45,500 on an average personal computer would take approximately 10,000 seconds (2.78 hours).

Benefits of Keyword Upweighting

In Figure 2, active learning did not improve the performance on the human rights corpus, and the F1 score was lower than other corpora in general. One reason for the early poor performance of *activeText* may be length of documents. Because each document of the human rights corpus consists of one sentence only, the average length is shorter than other corpora.[32] This means that the information the models can learn from labeled documents is less compared to the other corpora with longer documents. In situations like this, providing keywords in addition to document labels can improve classification performance because it directly shifts the values of the word-class probability matrix, $\boldsymbol{\eta}$, even when the provided keywords is not in the already labeled documents.

Figure 4 compares the performance with and without providing keywords. The darker lines show the results with keywords and the lighter lines without. The columns specify the proportion of documents associated with the class of interests: 5%, 50% and the population proportion (16%). As in the previous exercises, 20 documents are labeled at each sampling step, and 100 Monte Carlo simulations are performed to stabilize the randomness due to the initial set of documents to be labeled. We simulated the process of a user starting with no keywords for either class, and then being queried with extreme words indexed by v whose $\eta_{vk}/\eta_{vk'}$ is the highest for each class k, with up to 10 keywords for each class being chosen

Transformers and PyTorch libraries for the Python programming language and used the trained model to predict the labels for the remaining documents for each corpus.

[32]With the population data, the average length of each document is 121 (BBC), 17 (Wikipedia), 1620 (Supreme Court), and 9 (Human Rights)

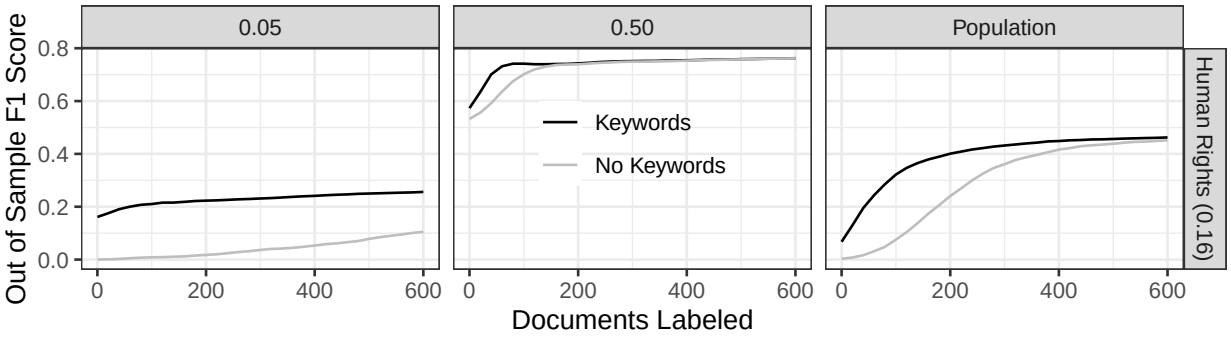

Figure 4: **Classification Results of *activeText* with and without Keywords**

based on the estimated $\boldsymbol{\eta}$ at a given iteration of the active process. To determine whether a candidate keyword should be added to the list of keywords or not, our simulated user checked whether the word under consideration was among the set of most extreme words in the distribution of the 'true' $\boldsymbol{\eta}$ parameter, which we previously estimated by fitting our mixture model with the complete set of labeled documents.[33]

The results suggest that providing keywords improves the performance when the proportion of documents is markedly imbalanced across classes. The keywords scheme improved the performance when the number of labeled documents is smaller on the corpus with 5% or 16% (population) labels associated with the class of interest. By contrast, it did not on the corpus where both classes were evenly balanced. These results highlight that our active keyword approach benefits the most when the dataset suffers from serious class-imbalance problems.[34]

One caveat is that we provided 'true' keywords, in the sense that we used the estimated $\boldsymbol{\eta}$ from a fully labeled dataset. In practice, researchers have to decide if candidate keywords are indeed keywords using their substantive knowledge. In this exercise, we believe that the keywords supplied to our simulation are what researchers with substantive knowledge about physical integrity rights can confidently adjudicate. For example, the keywords, such as "torture," "beat," and "murder," match our substantive understanding of physical integrity right violation. Nevertheless, humans can make mistakes, and some words may be difficult to judge. Thus, we examined the classification performance with varying degrees in the amount of error at the keyword labeling step. In SI H.1, we show that the active keyword approach still improves the classification performance compared to the no-keyword approach – even

[33]Specifically, the simulated user checked whether the word in question was in the top 10% of most extreme words for each class using the 'true' $\boldsymbol{\eta}$ parameter. If the candidate word was in the set of 'true' extreme words, it was added to the list of keywords and upweighted accordingly in the next active iteration.

[34]SI G demonstrates how active keyword works by visualizing the word-class matrix, $\boldsymbol{\eta}$, at each active iteration.

in the presence of small amounts (less than 20%) of "honest" (random) measurement error in keyword labeling.

Reanalysis with Fewer Human Annotations

To further illustrate our proposed approach for text classification, in this section, we reanalyze the results in Gohdes (2020) and Park et al. (2020). We show that via *activeText*, we arrive at the same substantive conclusions advanced by these authors but using only a small fraction of the labeled data they originally used.

Internet Accessibility and State Violence (Gohdes, 2020)

In the article "Repression Technology: Internet Accessibility and State Violence," Gohdes (2020) argues that higher levels of Internet accessibility are associated with increases in targeted repression by the state. The rationale behind this hypothesis is that through the rapid expansion of the Internet, governments have been able to improve their digital surveillance tools and target more accurately those in the opposition. Thus, even when digital censorship is commonly used to diminish the opposition's capabilities, Gohdes (2020) claims that digital surveillance remains a powerful tool, especially in areas where the regime is not fully in control.

To measure the extent to which killings result from government targeting operations, Gohdes (2020) collects 65,274 reports related to lethal violence in Syria. These reports contain detailed information about the person killed, date, location, and cause of death. The period under study goes from June 2013 to April 2015. Among all the reports, 2,346 were hand-coded by Gohdes, and each hand-coded report can fall under one of three classes: 1) government-targeted killing, 2) government-untargeted killing, and 3) non-government killing. Using a document-feature matrix (based on the text of the reports) and the labels of the hand-coded reports, Gohdes (2020) trained and tested a state-of-the-art supervised decision tree algorithm (extreme gradient boosting, `xgboost`). Using the parameters learned at the training stage, Gohdes (2020) predicts the labels for the remaining reports for which the hand-coded labels are not available. For each one of the 14 Syrian governorates (the second largest administrative unit in Syria), Gohdes (2020) calculates the proportion of biweekly government targeted killings. In other words, she collapses the predictions from the classification stage at the governorate-biweekly level.

We replicate Gohdes (2020) classification tasks using *activeText*. In terms of data preparation, we adhere to the very same decisions made by Gohdes (2020). To do so, we use the same 2,346 hand-labeled reports (1,028 referred to untargeted killing, 705 to a targeted killing, and 613 a non-government killing) of which 80% were reserved for training and 20%

to assess classification performance. In addition, we use the same document-feature matrices.[35] As noted in Active Learning, because *activeText* selects (at random) a small number of documents to be hand-labeled to initialize the process, we conduct 100 Monte Carlo simulations and present the average performance across initializations. As in Validation Performance, we set $\lambda = 0.001$. The performance of *activeText* and `xgboost` is evaluated in terms of out-of-sample F1 score. Following the discussion in Active vs. Passive Learning, we stopped the active labeling process at the 30th iteration when the out-of-sample F1 score stopped increasing by more than 0.01 units (our pre-specified threshold). Table 2 presents the results[36]. Overall, we find that as the number of active learning steps increases, the classification performance of *activeText* is similar to the one in Gohdes (2020). However, the number of hand-labeled documents that are required by *activeText* is significantly smaller (around one-third) if compared to the ones used by Gohdes (2020).

Table 2: Classification Performance: Comparison with Gohdes (2020) results

			Ouf-of-sample F1 Score per class		
Model	Step	Labels	Untargeted	Targeted	Non-Government
activeText	0	20	0.715	0.521	0.800
	10	220	0.846	0.794	0.938
	20	420	0.867	0.828	0.963
	30	**620**	**0.876**	**0.842**	**0.963**
	40	820	0.879	0.845	0.961
Gohdes (2020)		1876	0.910	0.890	0.940

In social science research, oftentimes, text classification is not the end goal but a means to quantify a concept that is difficult to measure and make inferences about the relationship between this concept and other constructs of interest. In that sense, to empirically test her claims, Gohdes (2020) conducts regression analyses where the proportion of biweekly government targeted killings is the dependent variable and Internet accessibility is the main independent variable – both covariates are measured at the governorate-biweekly level. Gohdes (2020) finds that there is a positive and statistically significant relationship between Internet access and the proportion of targeted killings by the Syrian government. Using the predictions from *activeText*, we construct the main dependent variable and replicate the main regression analyses in Gohdes (2020).

Tables in SI J reports the estimated coefficients, across the same model specifications in

[35]Gohdes (2020) removed stopwords, punctuation, and words that appear in at most two reports, resulting in 1,342 features and a document-feature matrix that is 99% sparse. The median number of words across documents is 13.

[36]The values in the bottom row are based on Gohdes (2020), Table A9.

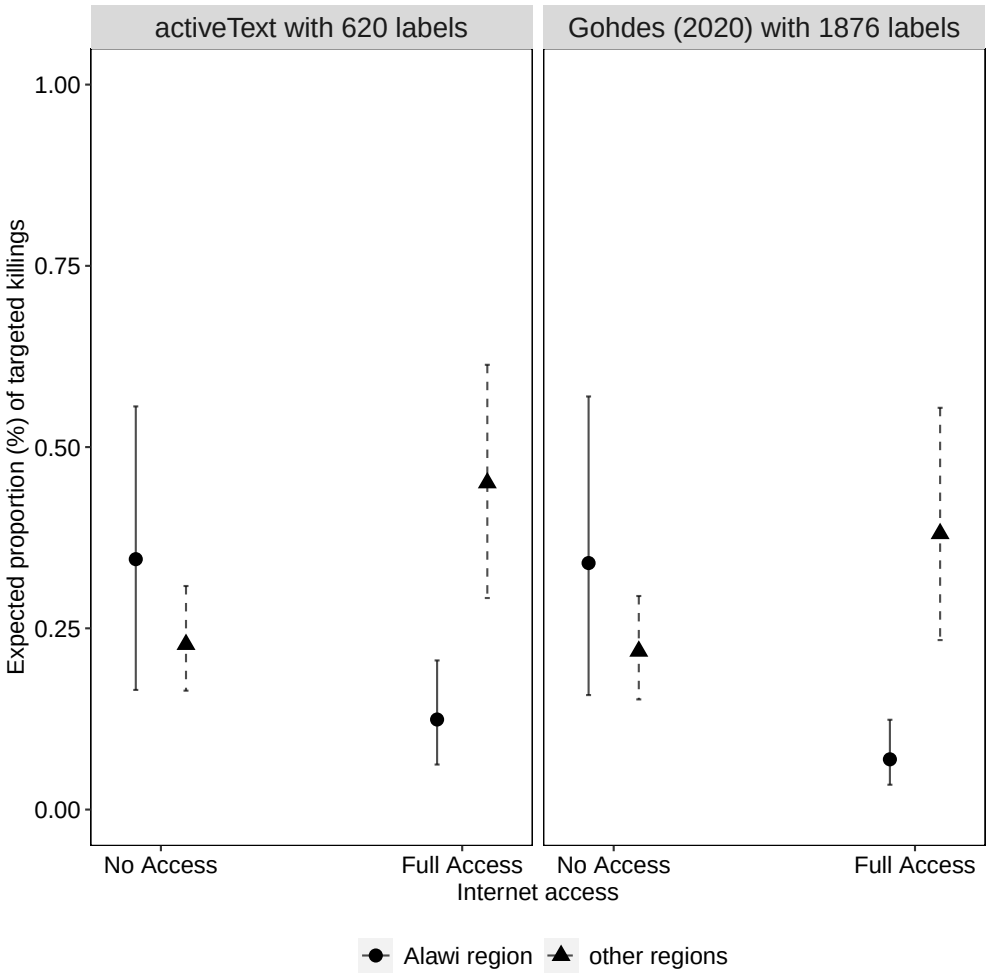

Figure 5: **Replication of Figure 3 in Gohdes (2020): Expected Proportion of Target Killings, Given Internet Accessibility and Whether a Region is Inhabitated by the Alawi Minority.** The results from *activeText* are presented in the left panel and those of Gohdes (2020) are on the right.

Gohdes (2020). The point estimates and the standard errors are almost identical whether we use `xgboost` or *activeText*. Moreover, Figure 5 presents the expected proportion of targeted killings by region and Internet accessibility. Gohdes (2020) finds that in the Alawi region (known to be loyal to the regime) when Internet access is at its highest, the expected proportion of targeted killings is significantly smaller compared to other regions of Syria. In the absence of the Internet, however, there is no discernible difference across regions (see Figure 5, right panel). Our reanalysis does not change the substantive conclusions by Gohdes (2020) (Figure 5, left panel), however, it comes just at a fraction of the labeling efforts (labeling 620 instead of 1876 reports). As noted above, these gains come from our active sampling scheme as it can select the most informative documents to be labeled.

Human Rights are Increasingly Plural (Park et al., 2020)

The question that drives the work of Park et al. (2020) is as follows: how the rapid growth (in the last four decades) of information communication technologies (ICTs) has changed the composition of texts referring to human rights? Park et al. (2020) make the observation that the average sentiment with which human rights reports are written has not drastically changed over time. Therefore, Park et al. (2020) advance the argument that if one wants to really understand the effect of changes in the access to information on the composition of human rights reports, it is necessary to internalize the fact that human rights are plural (bundles of related concepts). In other words, the authors argue that having access to new information has indeed changed the taxonomy of human rights over time, even when the tone has not.

To empirically test such a proposition, Park et al. (2020) conduct a two-step approach. First, via an SVM for text classification with three classes (negative, neutral, and positive sentiment), the authors show that the average sentiment of human rights reports has indeed remained stable even in periods where the amount of information available has become larger.[37] Second, they use a network modeling approach to show that while the average sentiment of these reports has remained constant over time, the taxonomy has drastically changed. In this section, using *activeText*, we focus on replicating the text classification task of Park et al. (2020) (which is key to motivating their puzzle).

As in the replication of Gohdes (2020), we adhere to the same pre-processing decisions made by Park et al. (2020) when working with their corpus of Country Reports on Human Rights Practices from 1977 to 2016 by the US Department of State. In particular, we use the same 4000 hand-labeled human rights reports (1182 are positive, 1743 are negative, and 1075 are neutral) and use the same document-feature matrices (which contain 30,000 features, a combination of unigrams and bigrams). Again, we conduct 100 Monte Carlo simulations and present the average performance across initializations. We stopped the active labeling process at the 25th iteration of our algorithm as the out-of-sample F1 score (from an 80/20 training/test split) does not increase by more than 0.01 units (see Figure K.1 in SI K).[38] Using the results from the classification task via *activeText*, the sentiment scores of 2,473,874 documents are predicted. With those predictions, we explore the evolution of the average

[37] As explained in Appendix A1 of Park et al. (2020), negative sentiment refers to text about a clear ineffectiveness in protecting or to violations of human rights; positive sentiment refers to text about clear support (or no restrictions) of human rights; and neutral sentiment, refers to stating a simple fact about human rights.

[38] The only point where we depart from Park et al. (2020) is that we use an 80/20 split for training/testing, while they use k-fold cross-validation. Conducting k-fold cross-validation for an active learning algorithm would require over-labeling and it would be computationally more expensive (the process should be repeated k times). Because of this difference we refrain from comparing our model performance metrics to theirs.

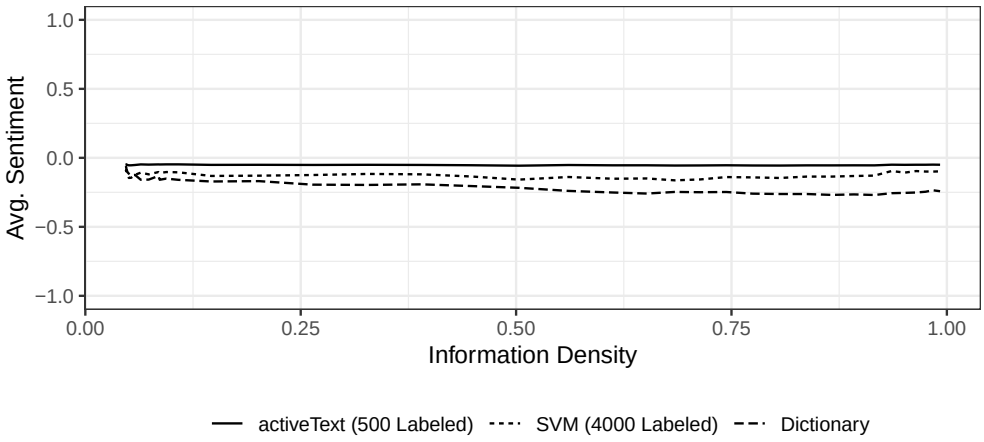

Figure 6: **Replication of Figure 1 in Park et al. (2020): The Relationship Between Information Density and Average Sentiment Score.**

sentiment of human rights reports per average information density score.[39]

Figure 6 shows that by labeling only 500 documents with *activeText*, instead of 4000 labeled documents used by Park et al. (2020) to fit their SVM classifier, we arrive at the same substantive conclusion: the average sentiment of human rights reports has remained stable and almost neutral over time. In Figure K.2 of SI K, we also show that this result is not an artifact of our stopping rule and it is robust to the inclusion of additional label documents (e.g, labeling 1000, 1500, and 2000 documents instead of just 500).

Discussion

Tuning the value of λ

As noted above, we downweight the information from unlabeled documents as we typically have more unlabeled than labeled documents. Moreover, since the labeled documents have been classified by an expert, we want to rely more on the information they bring for prediction.

An important practical consideration is: how to select the value of λ that maximizes the performance. One possible approach would be to adopt popular model selection methods (e.g. cross-validation) to choose the appropriate λ value during the model initialization process.[40] However, cross-validation may not be practical when the labeled data is scarce (or absent at the beginning of the process). Using our active learning approach is particularly,

[39]Information density is a proxy for ICTs based on a variety of indicators related to the expansion of communications and access to information, see Appendix B in Park et al. (2020).

[40]Indeed, it may be beneficial to tune the lambda value *across* active learning iterations.

we have observed across a variety of applications that very small values (e.g., 0.001 or 0.01) work the best on the corpora we used (see SI E). However, more work is needed to clearly understand the optimality criteria needed to select λ. We leave this question for future research.

Labeling Error

While our empirical applications assume that labelers are correct, human labelers do make mistakes. In SI H, we examine how mislabeling keywords and documents affect classification performance. Our results show that, if compared to the no-keyword approach, a small amount of random noise (classical measurement error) on keyword labeling does not hurt the classification performance. In contrast, random perturbations from true document labels do hurt the classification performance. A promising avenue for future research should center on developing new active learning algorithms that assign labelers based on their labeling ability and/or are robust to more pervasive forms of labeling error (differential and non-differential measurement error). For instance, assigning the most competent labelers with the most uncertain or difficult documents and assigning the least competent labelers with easier documents can optimize the workload of the labelers. At the same time, we note that users may be able to improve the quality of human labeling by other means, such as polishing cateogry concepts and better training of coders, in practical settings.

Conclusion

Human labeling of documents is the most labor-intensive part of social science research that uses text data. For automated text classification to work, a machine classifier needs to be trained on the relationship between text features and class labels, and the labels in training data are given manually. In this paper we have described a new active learning algorithm that combines a mixture model and active learning to incorporate information from labeled and unlabeled documents and better select which documents to be labeled by a human coder. Our validation study showed that the proposed algorithm performed at least as well as state-of-the-art methods such as BERT while reducing computational costs dramatically. We replicated two published political science studies to show that our algorithm lead to the same conclusions as the original papers but needed much fewer labeled documents. In sum, our algorithm enables researchers to save their manual labeling efforts without sacrificing quality.

Machine learning techniques are becoming increasingly popular in political science, but the barrier to entry remains too high for researchers without a technical background to make use of advances in the field. As a result, there is an opportunity to democratize access

to these methods. Towards this, we continue to work towards publishing the R package *activeText* on CRAN. We believe that our model will provide applied researchers a tool that they can use to efficiently categorize documents in corpuses of varying sizes and topics.

References

Airoldi, E. M., Fienberg, S. E., and Skinner, K. K. (2007), "Whose ideas? Whose words? Authorship of Ronald Reagan's radio addresses," *PS: Political Science & Politics*, 40(3), 501–506.

Altschuler, M., and Bloodgood, M. (2019), Stopping Active Learning Based on Predicted Change of F Measure for Text Classification,, in *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pp. 47–54.

Bishop, C. M., and Lassarre, J. (2007), "Generative or Discriminative? Getting the Best of Both Worlds," *Bayesian Statistics*, 8, 3–24.

Boydstun, A. E. (2013), *Making the news: Politics, the media, and agenda setting* University of Chicago Press.

Catalinac, A. (2016), *Electoral reform and national security in Japan: From pork to foreign policy* Cambridge University Press.

Cohn, D., Atlas, L., and Ladner, R. (1994), "Improving generalization with active learning," *Machine Learning*, 15(2), 201–221.

Cordell, R., Clay, K. C., Fariss, C. J., Wood, R. M., and Wright, T. (2021), "Recording repression: Identifying physical integrity rights allegations in annual country human rights reports," *International Studies Quarterly*, .

Dasgupta, S. (2011), "Two Faces of Active Learning," *Theoretical Computer Science*, 412(19), 1767–1781.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society, Series B*, 39(1), 1–38.

Denny, M. J., and Spirling, A. (2018), "Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it," *Political Analysis*, 26(2), 168–189.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018), "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, .

Eshima, S., Imai, K., and Sasaki, T. (2020), "Keyword assisted topic models," *arXiv preprint arXiv:2004.05964*, .

Gohdes, A. R. (2020), "Repression technology: Internet accessibility and state violence," *American Journal of Political Science*, 64(3), 488–503.

Greene, K. T., Park, B., and Colaresi, M. (2019), "Machine learning human rights and wrongs: How the successes and failures of supervised learning algorithms can inform the debate about information effects," *Political Analysis*, 27(2), 223–230.

Grimmer, J., Roberts, M. E., and Stewart, B. M. (2022), *Text as data: A New Framework for Machine Learning and the Social Sciences* Princeton University Press.

Grimmer, J., and Stewart, B. (2013), "Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts.," *Political Analysis*, 21(3), 267–297.

Hanneke, S. (2014), "Theory of Disagreement-Based Active Learning," *Foundations and Trends in Machine Learning*, 7(2-3), 131–309.

Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning*, Springer Series in Statistics, New York, NY, USA: Springer New York Inc.

Hino, H. (2021), "Active Learning: Problem Settings and Recent Developments," *Journal of the Japan Statistical Society, Japanese Issue*, 50(2), 317–342.

Hoi, S., Jin, R., and Lyu, M. R. (2006), Large-Scale Text Categorization by Batch Mode Active Learning,, in *WWW 06: Proceedings of the 15th International Conference on World Wide Web, Edinburgh, Scotland, May 23*, Vol. 26, pp. 633–642.

Ishibashi, H., and Hino, H. (2020), Stopping criterion for active learning based on deterministic generalization bounds,, in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, eds. S. Chiappa, and R. Calandra, Vol. 108 of *Proceedings of Machine Learning Research*, PMLR, pp. 386–397.
URL: *https://proceedings.mlr.press/v108/ishibashi20a.html*

King, G., Pan, J., and Roberts, M. E. (2017), "How the Chinese government fabricates social media posts for strategic distraction, not engaged argument," *American political science review*, 111(3), 484–501.

Knox, D., Lucas, C., and Cho, W. K. T. (2022), "Testing Causal Theories with Learned Proxies," *Annual Review of Political Science*, 25(1), 419–441.

Lewis, D. D., and Gale, W. A. (1994), A Sequential Algorithm for Training Text Classifiers,, in *SIGIR '94*, eds. B. W. Croft, and C. J. van Rijsbergen, Springer London, London, pp. 3–12.

Lowande, K. (2018), "Who Polices the Administrative State?," *American Political Science Review*, 112(4), 874–890.

Lowande, K. (2019), "Politicization and Responsiveness in Executive Agencies," *The Journal of Politics*, 81(1), 33–48.

Miller, B., Linder, F., and Mebane, W. R. (2020), "Active Learning Approaches for Labeling Text: Review and Assessment of the Performance of Active Learning Approaches," *Political Analysis*, pp. 1–20.

Miller, D. J., and Uyar, H. (1996), A Mixture of Experts Classifier with Learning Based on Both Labelled and Unlabelled Data,, in *Advances in Neural Information Processing Systems*, eds. M. Mozer, M. Jordan, and T. Petsche, Vol. 9, MIT Press.

Motolinia, L. (2021), "Electoral Accountability and Particularistic Legislation: Evidence from an Electoral Reform in Mexico," *American Political Science Review*, 115(1), 97–113.

Ng, A., and Jordan, M. (2001), "On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes," *Advances in neural information processing systems*, 14.

Nielsen, R. A. (2017), *Deadly clerics: Blocked ambition and the paths to jihad* Cambridge University Press.

Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. (2000), "Text classification from labeled and unlabeled documents using EM," *Machine learning*, 39(2-3), 103–134.

Park, B., Greene, K., and Colaresi, M. (2020), "Human Rights are (Increasingly) Plural: Learning the Changing Taxonomy of Human Rights from Large-scale Text Reveals Information Effects," *American Political Science Review*, 114(3), 888–910.

Pennington, J., Socher, R., and Manning, C. D. (2014), Glove: Global vectors for word representation,, in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peterson, A., and Spirling, A. (2018), "Classification accuracy as a substantive quantity of interest: Measuring polarization in westminster systems," *Political Analysis*, 26(1), 120–128.

Rodriguez, P. L., and Spirling, A. (2022), "Word embeddings: What works, what doesn't, and how to tell the difference for applied research," *The Journal of Politics*, 84(1), 101–115.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019), "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, .
URL: *https://arxiv.org/abs/1910.01108*

Settles, B. (2011), *Synthesis Lectures on Artificial Intelligence and Machine Learning : Active Learning* Morgan & Claypool Publishers.

Spirling, A. (2012), "US treaty making with American Indians: Institutional change and relative power, 1784–1911," *American Journal of Political Science*, 56(1), 84–97.

Stewart, B. M., and Zhukov, Y. M. (2009), "Use of force and civil–military relations in Russia: an automated content analysis," *Small Wars & Insurgencies*, 20(2), 319–343.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017), "Attention is all you need," *Advances in neural information processing systems*, 30.

Yang, Y., Ma, Z., Nie, F., Chang, X., and Hauptmann, A. G. (2015), "Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization," *International Journal of Computer Vision*, 113, 113–127.