Improving Probabilistic Models in Text Classification via Active Learning

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

Draft version: November 2, 2020

Abstract

In their quest to answer important questions, social scientists resort to multiple forms of data. This has lead to the development of new methods of analysis to understand the patterns in these data. Among these methods, automated text classification has become a standard tool for applied researchers. However, current approaches for text classification fail to take advantage of all the data at our disposal. We propose an active learning model for text classification, which combines labeled and unlabeled data and significantly improves performance at the cost of manually labelling a small number of documents. Using data from Wikipedia discussion pages, BBC News articles, and historical Supreme Court opinions, we show that by introducing information about the structure of unlabeled data, our model frequently outperforms models that only consider information from the labeled data.

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

[†]Ph.D. Student, Department of Political Science, University of Michigan. Email: `mcbosley@umich.edu`.

[‡]Ph.D. Student, 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`.

1 Introduction

As the amount and diversity of available information have rapidly increased, social scientists often resort to multiple forms of data to answer substantive questions. In particular, the use of text-as-data in cutting-edge social science research has exploded over the past decade.[1] Modern strategies to classify text documents are generally separated into supervised and unsupervised algorithms. Supervised approaches use associations between word frequencies and labels from a set of hand-coded documents to categorize documents, whereas unsupervised schemes separate the latent clusters of documents without needing to learn from a subset of labeled documents. Both of these methods have downsides, however: in the former, hand-coding documents is labor-intensive and costly, requires expert knowledge and reconciliation of disagreements between coders to ensure label validity; in the latter, the substantive interpretation of the categories discovered by the clustering process can be difficult, and performance is severely threatened when the underlying modeling assumptions are not met and/or when the data lacks the necessary structure such that a strong signal can be recovered.

One way to reduce the cost of hand-coding documents in the supervised approach is active learning. In contrast to the passive learning models, where the training set of documents is chosen randomly, active learning models use measures of label uncertainty to iteratively flag highly informative documents for the researcher to label. An active approach has been shown to be particularly helpful when the proportion of labels in a corpus is unbalanced. However, current implementations of active learning have only been used to augment supervised approaches. That is, in each iteration of an active learning algorithm, only labeled documents are used to train the supervised classifier that tells the researcher which documents should be labeled in the next active step.

Our innovation is to augment the active learning with unsupervised clustering in addition to the supervised approach. In this way, we provide a model that exploits the benefits of both the unsupervised and supervised approaches to improve the performance of text classifiers. Building on the work of Nigam et al. (2000) we use a probabilistic mixture model of text along with the expectation-maximization (EM) algorithm to combine the information from both labeled and unlabeled documents at each stage of the proposed active learning algorithm.

Specifically, we improve upon existing approaches in at least three ways. First, we show that our model outperforms simpler active learning method with supervised learning, when applied to various political science datasets. Second, we show that in some cases, allowing multiple latent clusters to be linked with one classification category improves the classification performance. Third, we develop the R package *activeText*, which provides fast and scalable implementation of our methods. The *activeText* package will allow researchers in a variety of fields to quickly and intuitively use the proposed methods to accurately classify large collections of text.

We believe that our approach can substantially benefit different research communities. From

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

providing innovative ways to minimize the amount of hand-coding and improve the precision of probabilistic models, to building tools that can be easily used and accessed, our overall goal with this project is to facilitate the research pipeline of other scientists.

This paper proceeds as follows. In Section 2 we describe the active learning process, and derive the probabilistic model we use at each stage of the algorithm. In Section 3, we implement our active learning algorithm on three datasets, and show how under certain conditions our model outperforms active versions of the Support Vector Machines and Naive Bayes classification algorithms. Finally, in Section 4 we summarize our results and discuss next steps.

2 The Method

This section describes our method for text classification. For the probabilistic model at the heart of the algorithm, we build on the work of Nigam et al. (2000), who show that a Naive Bayes classification algorithm can be augmented by including information about the clustering of unlabeled data. We insert our model into an active learning framework, at each step estimating the parameters of a probabilistic mixture model of text using the expectation-maximization (EM) algorithm to combine information from both labeled and unlabeled documents. In doing so, we build upon the work of Miller et al. (2020) by adding a semi-supervised component to the active learning process.

In Section 2.1 we provide an overview of our model. In Section 2.2 we derive our probabilistic model and show how its parameters are estimated by the EM algorithm. In Section 2.3, we describe the active learning algorithm.

2.1 Overview

In machine learning, semi-supervised learning aims to take advantage of the ideas behind supervised and unsupervised learning. Specifically, in semi-supervised learning, labeled and unlabeled data are used to learn a better classification rule than one based on labeled or unlabeled data alone. However, the most notorious problem with the implementation of semi-supervised approaches is the scarcity of labeled data. Due to this imbalance problem, for any classifier to be able to extract signal from labeled data and not be informed by unlabeled data alone, it is key to device ways to increase the relative importance of the labeled data. In the realm of text classification tasks, Nigam et al. (2000) proposed a solution to this problem. In their model, unlabeled and labeled data are combined to classify text into different classes, but the model adjusts how much it learns from the unlabeled data. In other words, the model balances the relative importance of each type of data, and by doing so, the classifier can learn from the labeled data as well – even if the vast majority of the data is unlabeled.

The main disadvantage of the method advanced by Nigam et al. (2000) is that pre-existing labeled data is required to train and test the classifier. However, in practice, only in exceptional cases a researcher will have access to labeled data in advance. Moreover, labeling data is considered an expensive and time-consuming task, thus even if a researcher is willing to obtain labeled data, there is no guidance on how to obtain the most informative data to be labeled such that the process is more efficient.

To overcome these challenges, we propose an active learning approach for probabilistic text classification.[2] In our method, labels are obtained when the classifier queries a human about a small sample of cases where it has problems to adjudicate labels. Furthermore, our classifier can be initially trained using unlabeled data only and subsequently, incorporate labeled data into the process. Thus, no pre-existing labeled data is needed and by focusing its attention on the most difficult to classify cases, it provides guidance on how to efficiently label data.

2.2 Model

We present our model and the observed document likelihood in Section 2.2.1, and show how we use the EM algorithm to estimate the class and word probability parameters in 2.2.2.

2.2.1 Probabilistic Model

Let D be a $N \times V$ document feature matrix, where N is the number of documents, and V is the size of features. We use a binary $N \times 2$ matrix, C, to indicate the class of each document. If a document i is assigned to the negative class, $C_{i1} = 1$ and $C_{i2} = 0$. If it is assigned to the positive class, $C_{i1} = 0$ and $C_{i2} = 1$. We use a $N \times K$ matrix, Z to represent the latent cluster, where K is the number of latent dimensions. If a document i is assigned to the k th cluster, $Z_{ik} = 1$ and $Z_{ik'} = 0, k' \neq k$.

We first draw a prior of being assigned to each cluster $\pi \in \mathcal{R}_+^K$ from Dirichlet distribution. Given π, we draw the latent cluster for each document $Z_i, i = 1 \ldots N$ from Multinomial distribution. We also draw a prior for each word being assigned to each cluster $\eta \in \mathcal{R}_+^{V \times K}$ from Dirichlet distribution. Given η and Z, we draw words from Multinomial distribution. The following describes our generative model.

$$\begin{aligned} \pi &\sim Dir(\alpha) \\ Z_{i\cdot} &\overset{i.i.d}{\sim} Multinomial(1, \pi) \\ \eta_{\cdot k} &\overset{i.i.d}{\sim} Dir(\beta) \\ D_{i\cdot} | Z_{ik} = 1 &\overset{i.i.d}{\sim} Multinomial(\eta_{\cdot k}) \end{aligned} \tag{1}$$

When the number of latent cluster is 2, the latent cluster and the class has one-to-one correspondence. ($C_{ik} = 1$ if $Z_{ik} = 1, k = 1, 2$). If the number of cluster is more than two, we assume that only one of the clusters are associated with the positive class and the rest of the clusters are associated with the negative class.

The observed likelihood is the following. Let D^{lp}, D^{ln} and D^{u} be the set of document feature

[2]See Settles (2010) for a great introduction to active learning.

matrix for positive labeled documents, labeled documents, and negative documents, respectively. Likewise, C^{lp} and C^{ln} be the set of positive and negative labels.

$$
\begin{aligned}
&p(\pi, \eta | D, C^l) \\
&\propto p(\pi)p(\eta)p(D^{lp}, C^{lp}|\pi,\eta)p(D^{ln}, C^{ln}|\pi,\eta)\Big[p(D^u|\pi,\eta)\Big]^\lambda \\
&= p(\pi)p(\eta) \times \prod_{i=1}^{N^{lp}} p(D_i^{lp}|Z_{ik^*},\eta)p(Z_{ik^*}|\pi) \\
&\quad \times \prod_{i=1}^{N^{ln}} \sum_{k\neq k^*} \Big\{p(D_i^{ln}|Z_{ik},\eta)p(Z_{ik}|\pi)\Big\} \times \left[\prod_{i=1}^{N^u}\sum_{k=1}^{K}\Big\{p(D_i^u|Z_{ik},\eta)p(Z_{ik}|\pi)\Big\}\right]^\lambda \qquad (2)\\
&\propto \underbrace{\prod_{k=1}^{K}\pi_k^{\alpha_k-1}\prod_{v=1}^{V}\prod_{k=1}^{K}\eta_{vk}^{\beta_k-1}}_{\text{prior}} \times \underbrace{\prod_{i=1}^{N^{lp}}\Big[\prod_{v=1}^{V}\eta_{vk^*}^{D_{iv}} \times \pi_{k^*}\Big]}_{\text{positive labeled doc. likelihood}} \\
&\quad \times \underbrace{\prod_{i=1}^{N^{ln}}\sum_{k\neq k^*}\Big\{\prod_{v=1}^{V}\eta_{vk}^{D_{iv}} \times \pi_k\Big\}}_{\text{negative labeled doc. likelihood}} \times \underbrace{\left[\prod_{i=1}^{N^u}\sum_{k=1}^{K}\Big\{\prod_{v=1}^{V}\eta_{vk}^{D_{iv}} \times \pi_k\Big\}\right]^\lambda}_{\text{unlabeled doc. likelihood}}
\end{aligned}
$$

We weight the term for the unlabeled dcoument by $\lambda \in (0, 1)$. This is because we typically have much more unlabeled document than labeled documents. By downweighting the infomration from the unlabeled document, we can use more reliable information from labeled documents than from unlabeled documents.

2.2.2 Parameter Estimation with the EM Algorithm

We estimate the parameters π and η using EM algorithm Dempster et al. (1977).[3] Taking the expectation of the log complete likelihood function (Q function),

$$
\begin{aligned}
Q \equiv \mathbb{E}_{Z|\pi,\eta,D,C}[p(\pi,\eta,Z|D,C)] &= \sum_{k=1}^{K}(\alpha_k-1)\log\pi_k + \sum_{v=1}^{V}\sum_{k=1}^{K}(\beta_k-1)\eta_{vk} \\
&+ \sum_{i=1}^{N^{lp}}\Big[\big(\sum_{v=1}^{V}D_{iv}\log\eta_{vk^*}\big) + \log\pi_{k^*}\Big] \\
&+ \sum_{i=1}^{N^{ln}}\sum_{k\neq k^*}p_{ik}\Big\{\sum_{v=1}^{V}D_{iv}\log\eta_{vk} + \log\pi_k\Big\} \qquad (3)\\
&+ \lambda\sum_{i=1}^{N^u}\sum_{k=1}^{K}p_{ik}\Big\{\sum_{v=1}^{V}D_{iv}\log\eta_{vk} + \log\pi_k\Big\}
\end{aligned}
$$

[3]Our implementation is presented as pseudocode in Algorithm 1.

Algorithm 1: EM algorithm to classify text

```
Result: Maximize $p(\pi^{(t)}, \eta^{(t)} \mid D^l, Z^l, D^u)$
if In the first iteration of Active learning then
    Initialize $\pi$ and $\eta$ by Naive Bayes;
        $\pi^{(0)} \leftarrow \mathrm{NB}(D^l,\ Z^l)$;
        $\eta^{(0)} \leftarrow \mathrm{NB}(D^l,\ Z^l)$;
else
    Inherit $\pi^{(0)}$ and $\eta^{(0)}$ from the preivous iteration of Active learning;
end
while $p(\pi^{(t)}, \eta^{(t)} \mid D^l, Z^l, D^u)$ does not converge do
    (1) E step: obtain the probability of the class for unlabeled documents;
        $p(Z^u \mid \pi^{(t)}, \eta^{(t)} D^l, Z^l, D^u) \leftarrow \text{E step}(D^u,\ \pi^{(t)},\ \eta^{(t)})$;
    (2) Combine the estimated classes for the unlabeled docs and the known classes for
     the labeled docs;
        $p(Z \mid \pi^{(t)}, \eta^{(t)}, D^l, Z^l, D^u) \leftarrow \text{combine}(D^l,\ D^u,\ Z^l,\ p(Z^u \mid \pi^{(t)}, \eta^{(t)}, D^l, Z^l, D^u))$;
    (3) M step: Maximize $Q \equiv \mathbb{E}[p(\pi, \eta, Z^u \mid D^l, Z^l, D^u)]$ w.r.t $\pi$ and $\eta$;
        $\pi^{(t+1)} \leftarrow \text{argmax } Q$;
        $\eta^{(t+1)} \leftarrow \text{argmax } Q$;
    (4) Check convergence: Obtain the value of $p(\pi^{(t+1)}, \eta^{(t+1)} \mid D^l, Z^l, D^u)$;
end
```

where p_{ik} is the probability of a document i being assigned to the k th cluster. If a document has a positive label, it has to have the cluster associated with the positive class. Let this cluster be k^*. Then, $p_{ik^*} = 1$ and $p_{ik} = 0, k \neq k^*$. If a document has a negative label,

$$p_{ik} = \frac{\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k}{\sum_{k \neq k^*} \left[\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k\right]} \tag{4}$$

If a document has no label,

$$p_{ik} = \frac{\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k}{\sum_{k=1}^{K} \left[\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k\right]} \tag{5}$$

In the M-step, we maximize the Q function, and obtain the updating equations for π and η. The updating equation for π is the following.

$$\hat{\pi_k} \propto \begin{cases} \alpha_k - 1 + \sum_{i=1}^{N^{ln}} p_{ik} + \lambda \sum_{i=1}^{N^u} p_{ik} & \text{if } k \neq k^* \\ \alpha_k - 1 + N^{lp} + \lambda \sum_{i=1}^{N^u} p_{ik^*} & \text{if } k = k^* \end{cases} \tag{6}$$

where N^{ln} is the number of documents with negative label, and N^u is with no label and N^{lp} is the number of documents with positive label. Essentially, this counts the number of document that has or predicted to have the k th cluster. We downweight the information from unlabeled document by λ, to utilize more reliable information from labeled documents.

The updating equation for η is the following.

$$\hat{\eta}_{vk} \propto \begin{cases} (\beta_k - 1) + \sum_{i=1}^{N^{ln}} p_{ik} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{ik} D_{iv} & \text{if } k \neq k^* \\ (\beta_k - 1) + \sum_{i=1}^{N^{lp}} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{ik^*} D_{iv} & \text{if } k = k^* \end{cases} \tag{7}$$

This essentially counts the number of words in the document that has or is predicted to have k th cluster. We also downweight the information from the unlabeled documents by λ.

2.3 Active Learning

Our active learning algorithm[4] can be split up into the following distinct steps: *initialization* with the Naive Bayes algorithm, *estimation* of the probability that each unlabeled document belongs to the positive class, *selection* of the unlabeled documents that the model is most uncertain about, and *labeling* of the selected documents by a human coder. The algorithm then *iterates* until it is reaches one of a pre-specified set of thresholds.

2.3.1 Initialization

The model is *initialized* with a small number of labeled documents.[5] The information from these documents is used to train a Naive Bayes classification algorithm[6], which outputs a number-of-classes length vector of class probabilities π, where each element is the posterior probability that a document belongs to a given class, and a number-of-words by number-of-classes matrix of word probabilities η, where each element represents the posterior probability that a word corresponds to a given class. Using the word and class likelihood parameters, we generate a number-of-documents by number-of-classes matrix, where each value represents the likelihood that each of the unlabeled documents belongs to a given class.[7]

2.3.2 Estimation

Using the document class probabilities generated by the initialization step, the model re-estimates the η and π parameters. When the λ parameter is equal to 1, the model treats each row of the document-class probability matrix equally, regardless of whether the document labeled deterministically by a human, or probabilistically by the algorithm. As the λ parameter moves from 1 towards 0, the model increasingly down-weights the information that the probabilistically labeled documents contribute to the estimation of η and π, such that when λ is 0, the model *ignores* all information from the probabilistically labeled documents. Once the η and π parameters are re-estimated, the model then uses the parameter values to re-label the probabilistically labeled documents. The resulting document-class probability matrix is used to re-estimate η and π, which

[4]We describe the active learning process in pseudocode in Algorithm 2.

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

[6]This process is equivalent to an initializing M-step from the EM algorithm in Section 2.2.2.

[7]This is E-step from the EM algorithm in Section 2.2.2.

Algorithm 2: Active learning with EM algorithm to classify text

```
Result: Obtain the predicted classes of all documents at least with some certainty
Initialize $D^l_{old}$ by sampling some documents randomly, and have humans label them ;
Initialize $D^u \leftarrow D \setminus D^l_{old}$;
while Not all documents are classified with some certainty yet do
    (1) Predict labels for each document in $D^u$ using Algorithm 1;
    (2) Sample $k$ most uncertain documents in $D^u$ and have humans labels them;
        $D^l_{new} \leftarrow k$ most uncertain documents in $D^u$;
    (3) Update labeled and unlabeled documents;
        $D^l_{old} \leftarrow D^l_{old} \cup D^l_{new}$;
        $D^u \leftarrow D \setminus D^l_{old}$;
end
```

are then used to re-label the probabilistically labeled documents, and so on. This is the process mapped by the EM algorithm, and it continues until the there is a sufficiently small change to the η and π parameters after they are re-estimated.

2.3.3 Selection

Using the document-class probability matrix from the last step of the preceding algorithm, the model uses *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 to calculating the absolute distance from 0.5 for each document. If, for a given document, the model predicts that it is *equally likely* to belong to either class (i.e., the probability of belonging to each class is 0.5), we would say that the model is *maximally uncertain* about the class that the document belongs to. On the other hand, if the model predicts that a document has a probability of belonging to particular class of 1.0, we would say that the model is *maximally certain* about the class that the document belongs to. Using this criteria, the model ranks all probabilistically labeled documents in descending order of uncertainty. The n most uncertain documents are then selected for human labeling, where n is an exogenously imposed model parameter.

2.3.4 Labeling

The model then provides the documents selected for labeling in the previous step to a human coder, who reads each document and imputes the 'correct' label. The newly-labeled documents are then added to the set of human-labeled documents.

2.3.5 Iteration

With the newly labeled documents, the model re-runs the EM algorithm. Instead of initializing with a Naive Bayes step as in 2.3.2, the model takes the final η and π parameter values from the previous estimation step as the starting point. From this point, the fundamental loop of estimation, selection, and labeling continues until one of three stopping conditions are met: (1) the model runs out of unlabeled documents to label; (2) the remaining unlabeled documents do not meet a particular uncertainty threshold; or (3) the maximum number of allowed active steps is reached.

3 Empirical Applications

This section summarizes results from the application of our model to three datasets: internal forum conversations of Wikipedia editors, BBC News articles, and the text from United States Supreme Court decisions. We explain our decisions regarding pre-processing steps, model evaluation, and model specifications, followed by a detailed discussion of the results for each dataset.

Pre-processing Steps. We employ the same pre-processing step for each of the three datasets using the *R* package *Quanteda.*[8]. For each dataset, we construct a *document-feature matrix* (DFM), where each row is a document and each column is a feature. Each feature is a stemmed unigram. We remove stopwords, features that occur extremely infrequently, as well as all features under 4 characters.

Model Evaluation. We use 80 percent of each dataset for training our model, and hold out the remaining 20 percent for evaluation. In any given step of the active learning algorithm, the training data is further divided into labeled and unlabeled documents. At each stage of the active algorithms model performance is evaluated in term of out-of-sample F1 score, where the F1 score is the harmonic mean of accuracy and precision. The out-of-sample F1 score is calculated using the held out testing data.

We also vary the proportion of positively labeled datasets to generate five seperate versions of each dataset. Specifically, we evaluate versions where 5, 10, 20, and 50 percent of the documents have a true positive label. We also include a 'population' specification for reference, where the full dataset is used without manipulation.

Model Specifications. For each dataset, we compare a variety of specifications of our *activeText* against the active Support Vector Machines (SVM) model from Miller et al. (2020).[9] For the *activeText* models, we vary the λ parameter between 0, 0.001, and 0.01. When the value of λ is 0, the model *ignores all information from unlabeled data in the training set.*, and is equivalent to the canonical Naive Bayes. When $\lambda > 0$, the model *learns from both the labeled and unlabeled data*, with down-weighting of information from the unlabeled data decreasing in λ. When the value of λ is 1, the model *does not distinguish between labeled and unlabeled documents in the training set.*[10]

Additionally, we vary the number of clusters between 2 and 5. When there are two clusters, there is a one-to-one mapping between clusters and classes in the binary classification exercise.

[8]See `https://quanteda.io`

[9]Specifically, we use the *margin sampling* variation of their model, which they showed performed the best in their analysis. In order to facilitate a fair comparison with our models, we evaluate their model with our Quanteda-based DFMs, rather than the SciKit Learn-based matrices in the original analysis. As a result, the cross-validation-of-DFMs feature in their original analysis is omitted here.

[10]Because the $\lambda = 1$ model infrequently performs better than the other reported models, we withhold it from our analysis.

As we increase the number of clusters, we maintain a one-to-one mapping between the *positive cluster* and the *positive class*, but allow *additional clusters to be estimated for the negative class.*

In all specifications, we use *entropy sampling* to select the documents that will be labeled in each active learning iteration, arranging the unlabeled documents in descending order in terms of Shannon entropy, then selecting the top n documents.[11] Additionally, the reported results are the average of 30 Monte Carlo iterations for each model. In each Monte Carlo iteration, the model is randomly initialized with 20 documents and in each active iteration (see Algorithm 2) 20 additional documents are labeled.[12]

3.1 Wikipedia Toxic Comments Dataset

The Wikipedia Toxic Comments dataset is a dataset made up of conversations between Wikipedia editors in Wikipedia's internal forums. The dataset was made openly available as part of a Kaggle competition,[13] and was used as a principle dataset of investigation by Miller et al. (2020). The basic classification task is to label a given speech as toxic or not, where toxicity is defined as including harassment and/or abuse of other users.[14] The complete dataset is comprised of roughly 560,000 documents, roughly 10 percent of which are labeled as toxic.

Figure 1 compares the performance of *activeText* models to the active Naive Bayes and SVM model for the Wikipedia Toxic Comments dataset. Each cell represents a particular specification of the number of clusters used by the model (by the rows) and the share of positive labeled documents used to evaluate the algorithm (by the columns). In each cell, the x-axis tracks the number of active steps at which the model is evaluated, and the y-axis the corresponding out-of-sample F1 score. The colored lines in each cell represent models with different specifications of the λ parameter. The green line represents the $\lambda = 0$ model, and is equivalent to Naive Bayes. The teal and purple lines represent the $\lambda = 0.001$ and $\lambda = 0.01$ models, respectively. The red line represents the active SVM model from Miller et al. (2020), which is included for reference and is identical in all cells.

The gap between the red and green lines in each of the cells indicates that the active Naive Bayes model consistently outperforms the active SVM model. Additionally, the gap between the teal and green lines shows that that the $\lambda = 0.001$ *activeText* model outperforms both the Naive Bayes baseline and active SVM models across all specifications. The difference in model performance is particularly striking when the share of positively labeled documents is 20 percent or less. For example, when the share of positive labeled documents is 0.10 and the number of clusters used is 2, fifteen (320 total documents labeled) active steps are required to achieve an

[11] Versions of the EM-based models with random, rather than entropy, sampling are included in Appendix A for reference.

[12] In order to ensure a fair comparison with the active SVM model, the *activeText* and SVM models are initialized with the same random documents for each Monte Carlo iteration.

[13] See `https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge`

[14] While the dataset also contains finer gradation of 'types' of toxicity, we like Miller et al. (2020) stick to the binary toxic-or-not classification task.

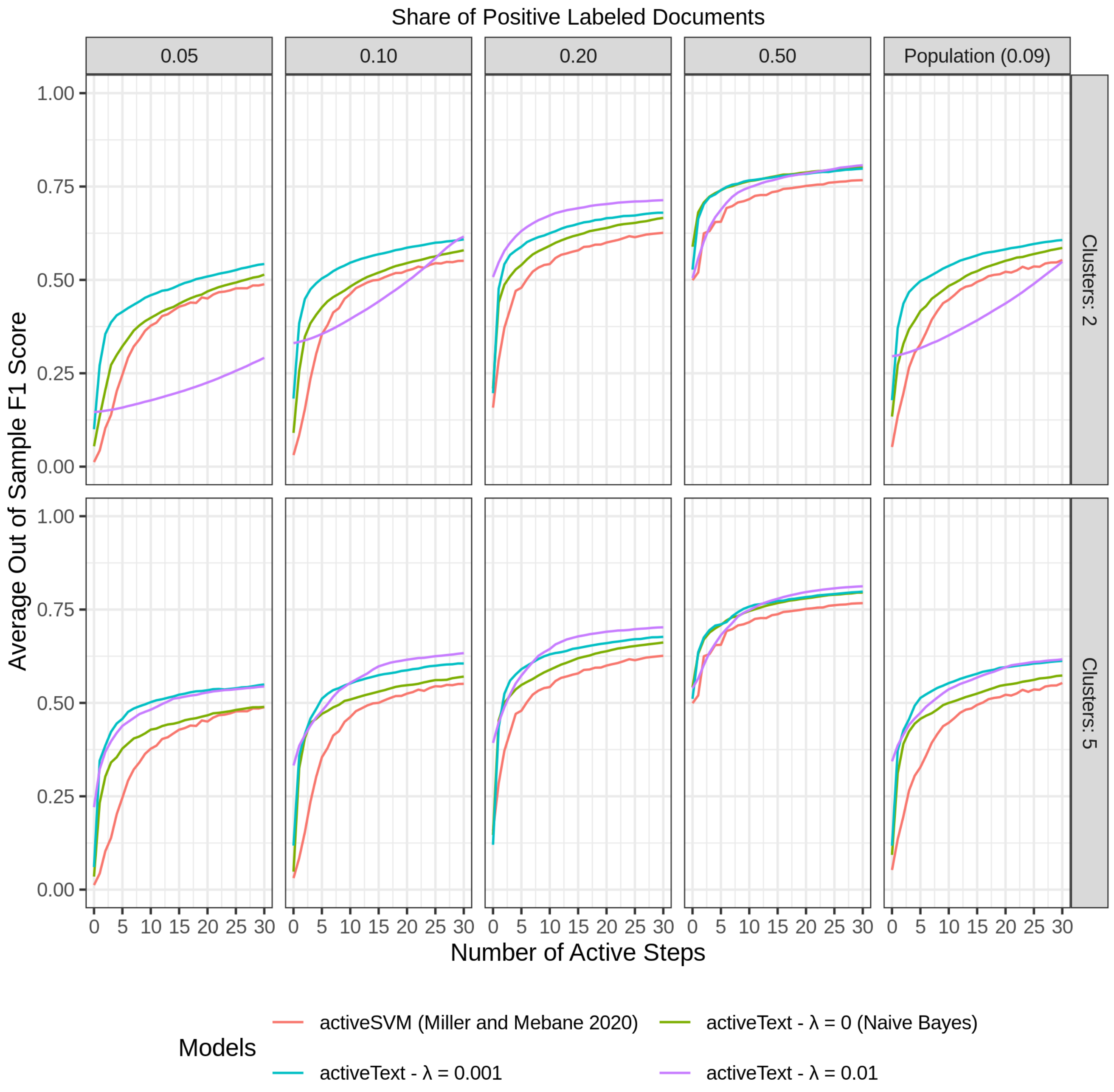

Figure 1: **Comparing the Out-of-Sample F1 Performance of Active Learning Models with the Wikitoxic Dataset.** The proportion of positive documents in the corpus varies across the column facets, and the number of clusters estimated varies across the row facets. Each active step labels an additional 20 documents, and each result is the average of 30 Monte Carlo iterations. The value of the λ parameter indicates the extent to which the model uses information from unlabeled documents. At $\lambda = 0$ ($\lambda = 1$) the model puts all (equal) weight on labeled (labeled and unlabeled) documents. In both the 2 and 5 cluster cases, the activeText models outperform both the active SVM model from Miller et al. (2020) and the Naive Bayes baseline across the board. This advantage is particularly notable when the proportion of positive labeled documents is low.

out-of-sample F1 score of 0.50 using the active SVM or active Naive Bayes model, whereas the *activeText* $\lambda = 0.001$ model achieves the same result in four active steps (100 total documents labeled.) While increasing the number of clusters from 2 to 5 does not improve the best models, it does notably improve the early performance of the purple line representing the $\lambda = 0.01$ model when the proportion of positive documents is 5 and 10 percent (i.e., in the first two columns).

3.2 BBC News Dataset

The BBC News Dataset is a collection of 2225 documents from the BBC news website, divided equally into five topics: business, entertainment, politics, sport, and technology. The classification exercise is to correctly predict whether or not an article belongs to the 'politics' topic.

Figure 2 compares the performance of *activeText* models to the active SVM and Naive Bates models for the BBC News dataset. Each cell represents a particular specification of the number of clusters used by the model (by the rows) and the share of positive labeled documents used to evaluate the algorithm (by the columns). In each cell, the x-axis tracks the number of active steps at which the model is evaluated, and the y-axis the corresponding out-of-sample F1 score. The colored lines in each cell represent models with different specifications of the λ parameter. The green line represents the $\lambda = 0$ model, and is equivalent to Naive Bayes. The teal and purple lines represent the $\lambda = 0.001$ and $\lambda = 0.01$ models, respectively. The red line represents the active SVM model from Miller et al. (2020), which is included for reference and is identical in all cells.

In the upper-left cell, where the share of positive labeled documents is 0.05 and the number of clusters is 2, the active SVM model (represented by the red line) outperforms both the active Naive Bayes model (represented by the green line) as well as both the $\lambda = 0.001$ and $\lambda = 0.01$ models (teal and purple lines, respectively). However, in all cells, the active SVM model is outperformed by the other models. While there is relatively little separation between the models when the number of clusters is 2 (the top row of cells), increasing the number of clusters to 5 (moving to the bottom row of cells) improves the results all *activeText* models, including the active Naive Bayes model. This improvement is particularly dramatic for the $\lambda = 0.01$ model (denoted by the purple line). In the third column, where the proportion of positive labeled documents is 0.20, the model achieves an out-of-sample F1 score of roughly 0.90 within the first two active steps when the other models need 10 steps.

3.3 Supreme Court Rulings Dataset

The Supreme Court Rulings dataset[15] is a collection of the text of 2000 US Supreme Court rulings between 1946 and 2012, divided into fourteen categories.[16] The classification exercise here is to correctly identify rulings that are categorized as 'criminal procedure'.

Figure 3 compares the performance of *activeText* models to the active SVM model for the

[15]See `http://www.supremecourtdatabase.org`.

[16]For a full list of categories, see `http://www.supremecourtdatabase.org/documentation.php?var=issueArea`.

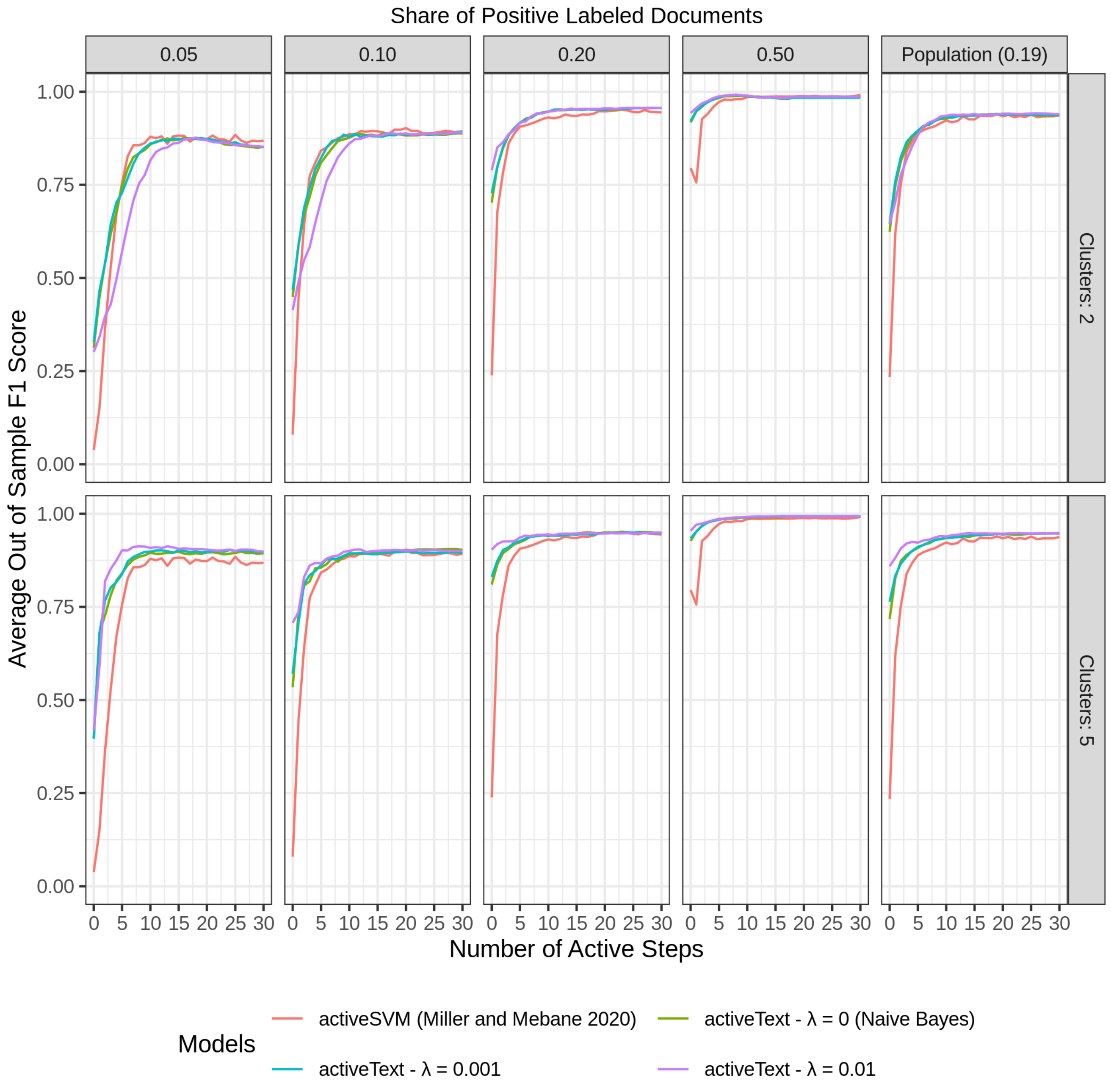

Figure 2: **Comparing the Out-of-Sample F1 Performance of Active Learning Models with the BBC News Dataset.** The proportion of positive documents in the corpus varies across the column facets, and the number of clusters estimated varies across the row facets. Each active step labels an additional 20 documents, and each result is the average of 30 Monte Carlo iterations. The value of the λ parameter indicates the extent to which the model uses information from unlabeled documents. At $\lambda = 0$ ($\lambda = 1$) the model puts all (equal) weight on labeled (labeled and unlabeled) documents. When only two clusters are used and the proportion of positive labeled documents is 0.05, the active SVM model from Miller et al. (2020) outperforms the activeText models. However, as the proportion increases, the activeText models perform better than the SVM model, particularly in the first few active steps. Moving from 2 to 5 clusters leads to an across the board improvement of the activeText models, with $\lambda = 0.01$ model notably outperforming both the active SVM and Naive Bayes baselines over the first 10 active steps.

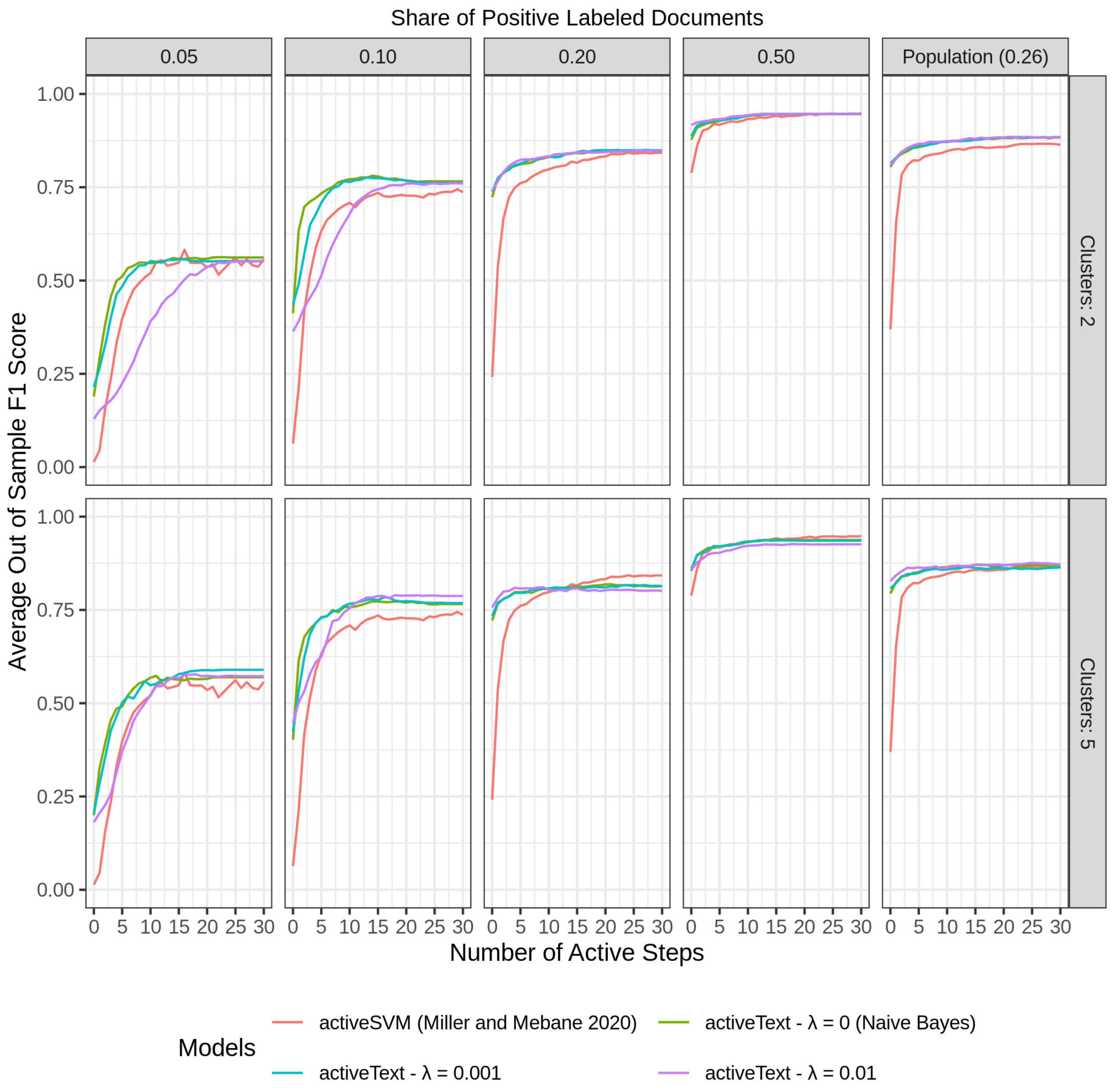

Figure 3: **Comparing the Out-of-Sample F1 Performance of Active Learning Models with the Supreme Courts Cases Dataset.** The proportion of positive documents in the corpus varies across the column facets, and the number of clusters estimated varies across the row facets. Each active step labels an additional 20 documents, and each result is the average of 30 Monte Carlo iterations. The value of the λ parameter indicates the extent to which the model uses information from unlabeled documents. At $\lambda = 0$ ($\lambda = 1$) the model puts all (equal) weight on labeled (labeled and unlabeled) documents. The $\lambda = 0$ model generally outperforms both the other activeText variants and the active SVM model from Miller et al. (2020). However, when 5 clusters are used and the share of positive labeled documents is 0.20 or greater, $\lambda = 0.01$ model outperforms the other models over the first 10 steps.

Supreme Court Rulings dataset. Each cell represents a particular specification of the number of clusters used by the model (by the rows) and the share of positive labeled documents used to evaluate the algorithm (by the columns). In each cell, the x-axis tracks the number of active steps at which the model is evaluated, and the y-axis the corresponding out-of-sample F1 score. The colored lines in each cell represent models with different specifications of the λ parameter. The green line represents the $\lambda = 0$ model, and is equivalent to Naive Bayes. The teal and purple lines represent the $\lambda = 0.001$ and $\lambda = 0.01$ models, respectively. The red line represents the active SVM model from Miller et al. (2020), which is included for reference and is identical in all cells.

The *activeText* models generally outperform the active SVM model (red line), regardless of the number of clusters.[17] When the share of positive labeled documents is 0.05 or 0.10 (the left-most two columns), the active Naive Bayes model (green line) outperforms the other models. As the share of positively labeled documents increases (moving right across the columns), the *activeText* models achieve almost increasingly similar results, while maintaining an advantage over the active SVM model (red line), particularly in the first few active iterations. Moving from 2 to 5 clusters (from the top to the bottom row) improves the early performance of the $\lambda = 0.01$ model (purple line), when the share of positive labeled documents is 0.05 or 0.10, it actually diminishes the performance of the other *activeText* models. When the share of positive labeled documents is 0.20, moving from 2 to 5 clusters provides a substantial boost to the $\lambda = 0.01$ model over the first three active iterations, causing it to reach an out-of-sample F1 score that the other models dont reach until 10 active steps.

4 Discussion and Conclusion

In this paper we have described a new active learning algorithm that combines information from labeled and unlabeled documents in order to better select which documents to be labeled by a human coder. We have shown that across three diverse datasets, our model almost always outperforms the active SVM algorithm from Miller et al. (2020), and that when we use the λ to appropriately down-weight the information the model learns from unlabeled data, we frequently outperform the active Naive Bayes baseline as well.

In the short-term, we see two improvements we can make to the model: on the model construction side, allowing the number of clusters for the positive class to be greater than one[18]; and on the model evaluation side, creating a more principled selection mechanism for the λ parameter. While we do not yet have a formal way of choosing the appropriate value of λ, we have observed across a variety of applications that very small values (e.g., 0.001 or 0.01) seem to work the best. It is possible, however, that we can adopt popular model selection methods (such as, for example,

[17]An exception is in the upper-left-most cell, where the $\lambda = 0.01$ model (purple line) is the worst performing model.

[18]Currently, when we allow the number of clusters to exceed the number of classes, we make an identification assumption that the positive class is associated with a single cluster, and map the remaining clusters to the negative class.

k-fold cross-validation) to choose the appropriate lambda value during the model initialization process.[19]

In the somewhat longer-term, we would like to implement a feature that allows the human labeler to specify which words they think are particularly likely to belong to one class rather than another. In this way, the human labeler would select both the class of the document that they are reading, and highlight a set of trigger words. These trigger words would then be up-weighted during the calculation of the η word probability matrix in the M-step of the EM algorithm, potentially boosting the speed at which the active learning algorithm learns to correctly distinguish between types of documents.

Machine learning techniques are becoming increasingly popular in Political Science, but frequently 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.

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

References

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 (Methodological)*, 39(1), 1–22.

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.

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.

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.

Settles, B. (2010), "Active Learning Literature Survey.,", Technical Report 2010-09-14, University Wisconsin–Madison.

A Appendix

Figures 4, 5, and 6 show the results of the EM-based models when using random sampling rather than active learning, as in Figures 1, 2, and 3. The SVM model from Miller et al. (2020) uses active learning across all implementations, for reference. As expected, using active learning drastically improves the results of the EM-based models when the proportion of positive labeled documents is low. As the proportion of positive labeled documents increases, the benefits from active learning diminish.

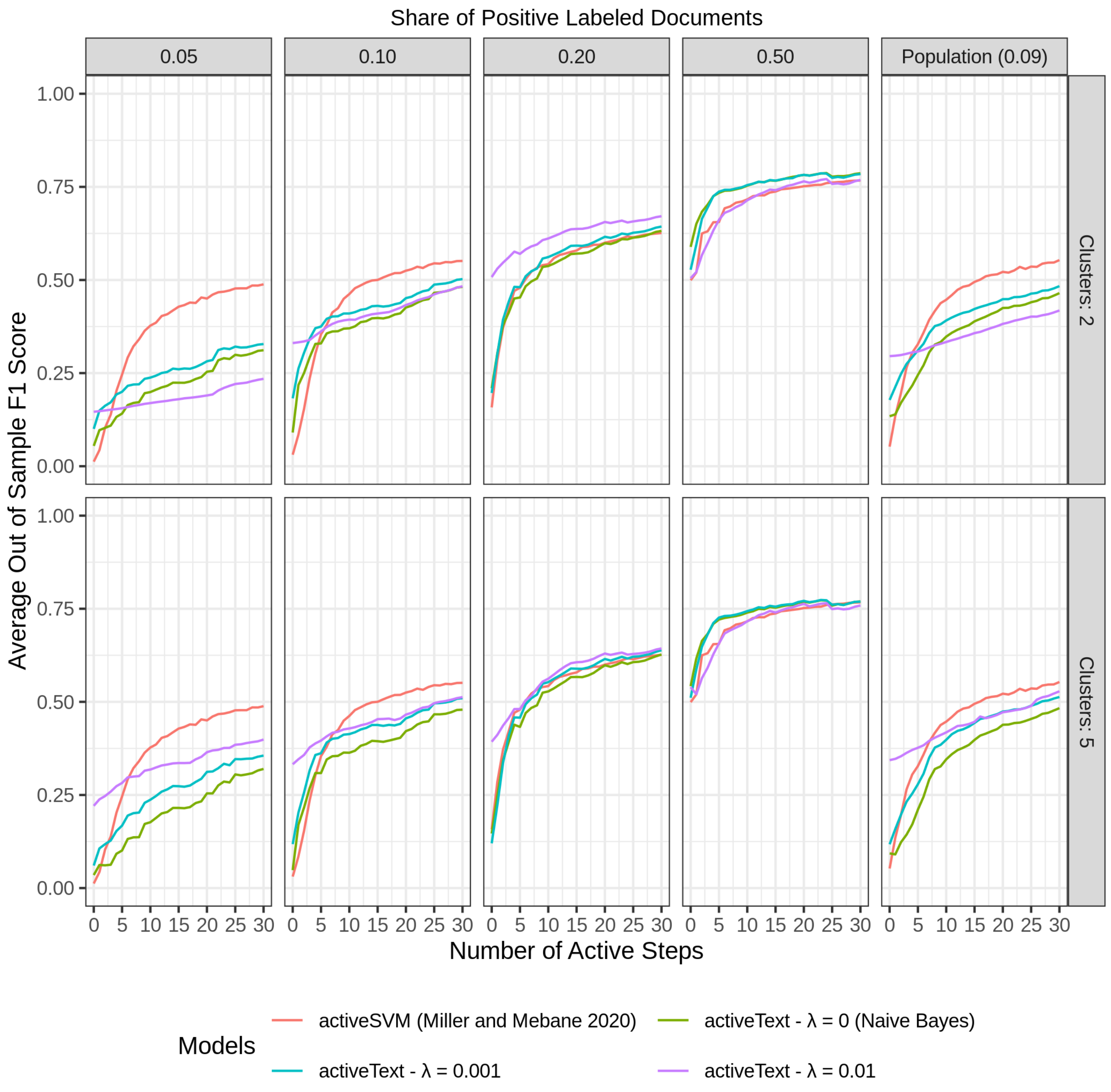

Figure 4: **Comparing the Out-of-Sample F1 Performance of Random Sampling Models with the Wikitoxic Dataset.**

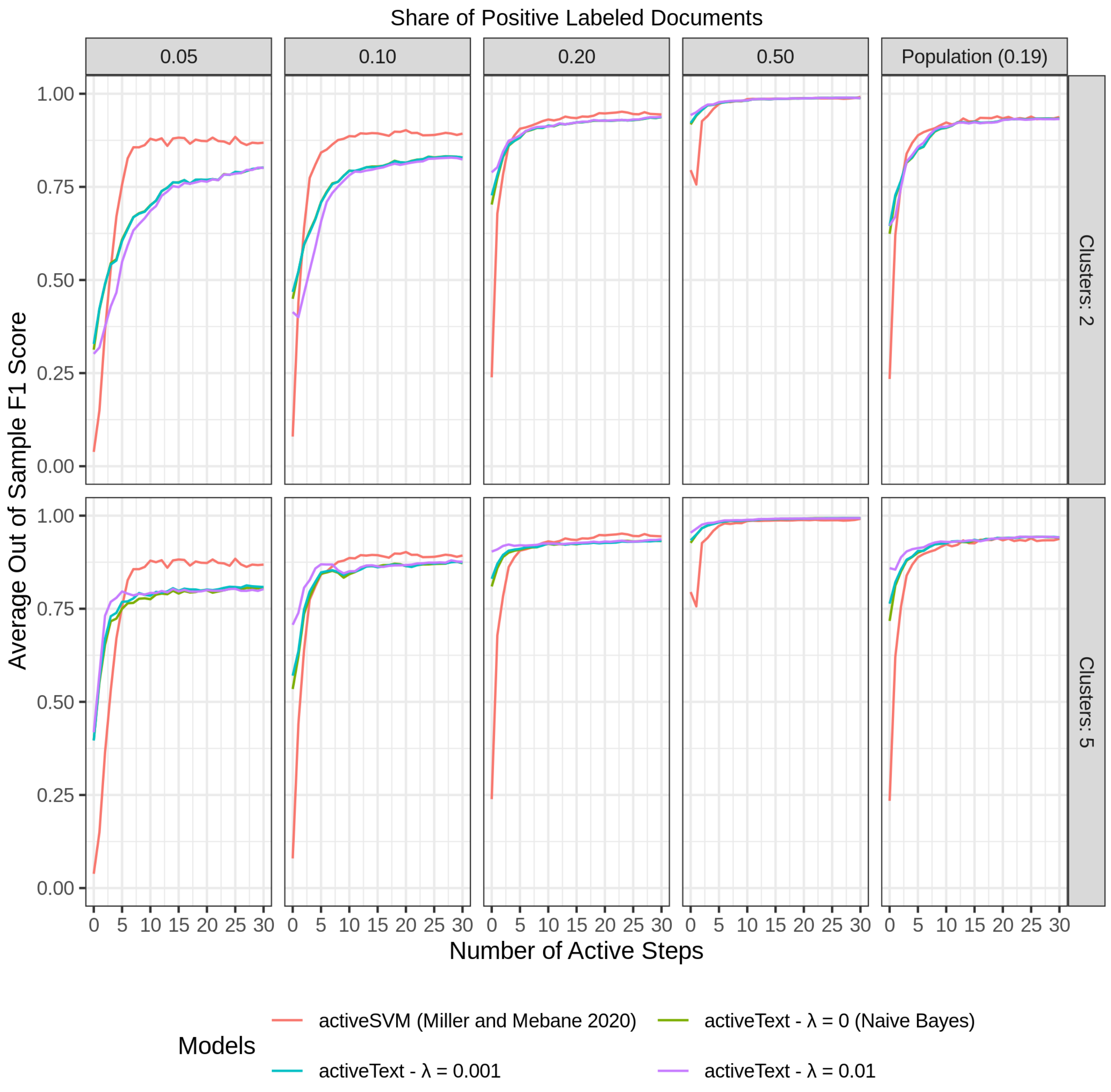

Figure 5: **Comparing the Out-of-Sample F1 Performance of Random Sampling Models with the BBC Dataset.**

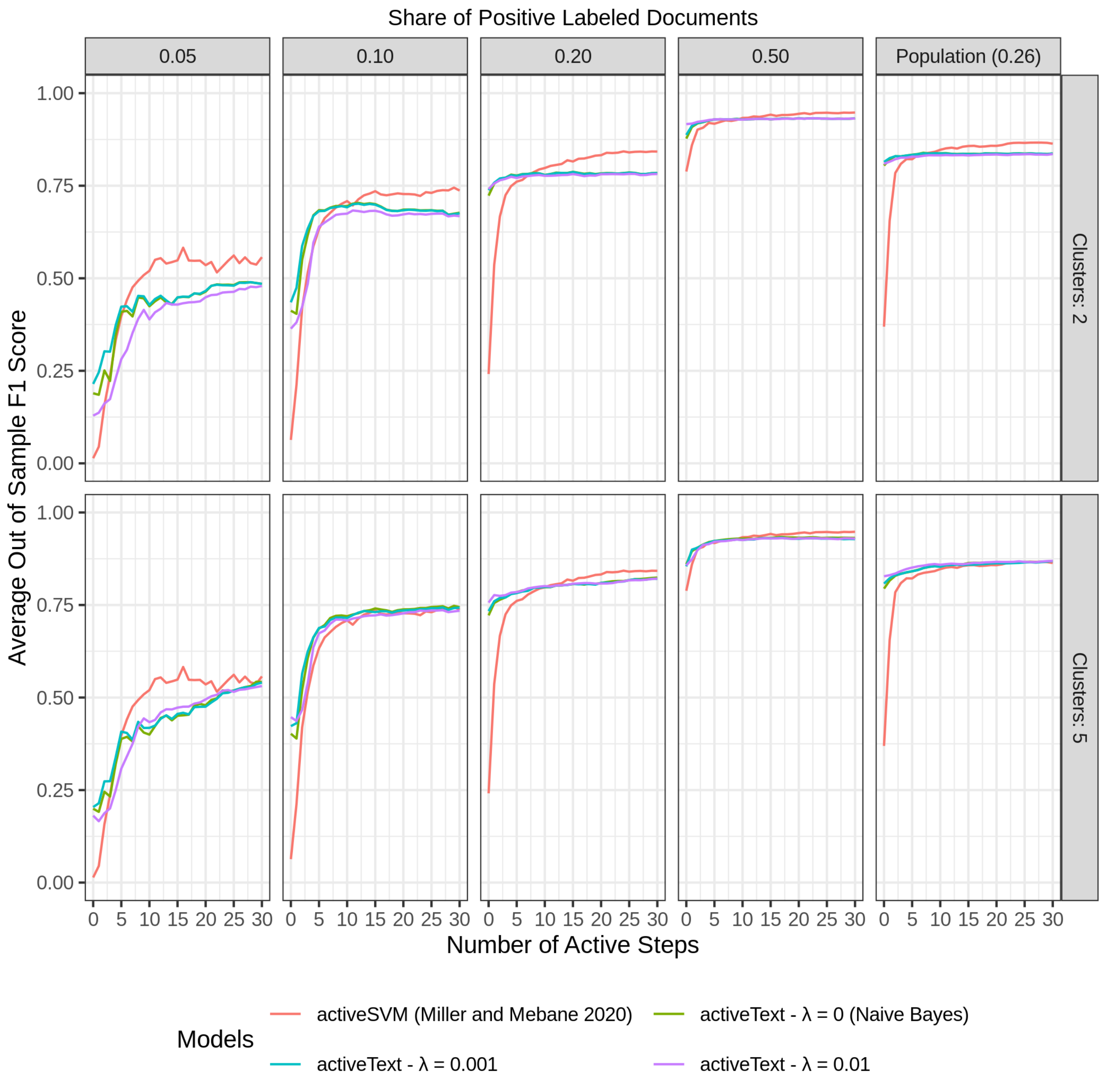

Figure 6: **Comparing the Out-of-Sample F1 Performance of Random Sampling Models with the Supreme Court Cases Dataset.**