Supplementary Information for Improving Probabilistic Models in Text Classification via Active Learning*

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

First draft: September 10, 2020
This draft: July 18, 2023

Contents

*We thank Ken Benoit, Yaoyao Dai, Chris Fariss, Yusaku Horiuchi, Kosuke Imai, Walter Mebane, Daichi Mochihashi, Kevin Quinn, Luwei Ying, 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, the 2022 Summer Meeting of the Japanese Society for Quantitative Political Science, and the 40th Annual Summer Meeting of the Society for Political Methodology, 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.

A Using Machine Learning for Text Classification

A.1 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. 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.*[1]

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) $\mathbf{D}$ 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.[2] Each element of the DFM encodes the frequency that a token occurs in a given document.[3] Once the researcher chooses how to encode their corpus as a matrix, she is left with a set of features corresponding to each document $\mathbf{D}$ and an unknown vector of true labels $\mathbf{Z}$, where each element of $\mathbf{Z}$ indicates whether a given document is political or not. Then, we can repose the classification question as follows: given $\mathbf{D}$, how might we best learn $\mathbf{Z}$, that is, whether each document is political or not?

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

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

[3] 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).

A.2 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 $\mathbf{Z}$ given the text data $\mathbf{D}$ i.e., directly estimate $p(\mathbf{Z}|\mathbf{D})$. In contrast, when using a generative model (e.g., Naive Bayes), learning the relationship between the $\mathbf{Z}$ and $\mathbf{D}$ is a two-step process. In the first step, the likelihood of the matrix of text data $\mathbf{D}$ and outcome labels $\mathbf{Z}$ is estimated given the data and a set of parameters θ that indicate structural assumptions about how the data is generated i.e., $p(\mathbf{D}, \mathbf{Z}|\theta)$ 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., $p(\mathbf{Z}|\mathbf{D}; \theta)$.

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.[4] This can help prevent overfitting when the amount of data in a corpus is small.[5] Conversely, because it is not necessary to model the data generating process directly, 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).[6]

A.3 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

[4]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.

[5]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 model has not seen before.

[6]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.

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

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

puts aside and label a set of randomly chosen documents that the active learning algorithm does not have access to.[7] 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 A.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 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

[7] 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*.

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.
Note that to facilitate exposition, in the main text, we use the words political and non-political labels to describe the problem of binary classification. Without loss of generality, in this supplemental information material, we use the positive vs. negative class dichotomy instead.

B Detailed explanations about the EM algorithm to estimate parameters

Let $\mathbf{D}^{lp}$, $\mathbf{D}^{ln}$ and $\mathbf{D}^{u}$ be the document feature matrices for documents with positive labels, documents with negative labels, and unlabeled documents, respectively. Also let N^{lp}, N^{ln}, and N^{u} be the number of documents with positive labels, negative labels, and documents without labels. Likewise, $\mathbf{C}^{lp}$ and $\mathbf{C}^{ln}$ be the vectors of positive and negative labels. Then, the observed-data likelihood is:

$$
\begin{aligned}
&p(\pi, \boldsymbol{\eta} | \mathbf{D}, \mathbf{C}^{lp}, \mathbf{C}^{ln}) \\
&\propto p(\pi) p(\boldsymbol{\eta}) p(\mathbf{D}^{lp}, \mathbf{C}^{lp} | \pi, \boldsymbol{\eta}) p(\mathbf{D}^{ln}, \mathbf{C}^{ln} | \pi, \boldsymbol{\eta}) \Big[p(\mathbf{D}^{u} | \pi, \boldsymbol{\eta}) \Big]^{\lambda} \\
&= p(\pi) p(\boldsymbol{\eta}) \times \prod_{i=1}^{N^{lp}} p(\mathbf{D}_i^{lp} | Z_i = 1, \eta) p(Z_i = 1 | \pi) \times \prod_{i=1}^{N^{ln}} \Big\{ p(\mathbf{D}_i^{ln} | Z_i = 0, \eta) p(Z_i = 0 | \pi) \Big\} \\
&\quad \times \left[\prod_{i=1}^{N^{u}} \Big\{ p(\mathbf{D}_i^{u} | Z_i = 1, \boldsymbol{\eta}) p(Z_i = 1 | \pi) + p(\mathbf{D}_i^{u} | Z_i = 0, \boldsymbol{\eta}) p(Z_i = 0 | \pi) \Big\} \right]^{\lambda} \\
&\propto \underbrace{\Big\{ (1-\pi)^{\alpha_0 - 1} \prod_{v=1}^{V} \eta_{v0}^{\beta_{0v} - 1} \Big\} \times \Big\{ \pi^{\alpha_1 - 1} \prod_{v=1}^{V} \eta_{v1}^{\beta_{1v} - 1} \Big\}}_{\text{prior}} \times \underbrace{\prod_{i=1}^{N^{lp}} \Big\{ \prod_{v=1}^{V} \eta_{v1}^{D_{iv}} \times \pi \Big\}}_{\text{positive labeled doc. likelihood}} \\
&\quad \times \underbrace{\prod_{i=1}^{N^{ln}} \Big\{ \prod_{v=1}^{V} \eta_{v0}^{D_{iv}} \times (1-\pi) \Big\}}_{\text{negative labeled doc. likelihood}} \times \underbrace{\left[\prod_{i=1}^{N^{u}} \Big\{ \prod_{v=1}^{V} \eta_{v0}^{D_{iv}} \times (1-\pi) \Big\} + \Big\{ \prod_{v=1}^{V} \eta_{v1}^{D_{iv}} \times \pi \Big\} \right]^{\lambda}}_{\text{unlabeled doc. likelihood}}
\end{aligned}
\tag{1}
$$

We weigh the part of the observed likelihood that refers to the unlabeled document with $\lambda \in [0, 1]$. This is done because we typically have many more unlabeled documents than labeled documents. By downweighting the information from the unlabeled document (i.e., setting λ to be small), we can use more reliable information from labeled documents than from unlabeled documents.

We estimate the parameters π and η using EM algorithm Dempster et al. (1977) and our implementation is presented as pseudocode in Algorithm 1. Note that by taking the

Algorithm 1: EM algorithm to classify text

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

expectation of the log complete likelihood function (Q function),

$$
\begin{aligned}
Q &\equiv \mathbb{E}_{\mathbf{Z}|\pi^{(t)},\boldsymbol{\eta}^{(t)},D,C}[p(\pi, \boldsymbol{\eta}, \mathbf{Z}|\mathbf{D}, \mathbf{C})] \\
&= (\alpha_0 - 1)\log(1-\pi^{(t)}) + (\alpha_1 - 1)\log \pi^{(t)} + \sum_{v=1}^{V}\Big\{(\beta_{0v}-1)\log \eta_{v0}^{(t)} + (\beta_{1v}-1)\log \eta_{v1}^{(t)}\Big\} \\
&\quad + \sum_{i=1}^{N^{lp}}\Big\{\sum_{v=1}^{V} D_{iv}\log \eta_{v1}^{(t)} + \log \pi^{(t)}\Big\} + \sum_{i=1}^{N^{ln}}\Big\{\sum_{v=1}^{V} D_{iv}\log \eta_{v0}^{(t)} + \log(1-\pi^{(t)})\Big\} \\
&\quad + \lambda\left[\sum_{i=1}^{N^u} p_{i0}\Big\{\sum_{v=1}^{V} D_{iv}\log \eta_{v0}^{(t)} + \log(1-\pi^{(t)})\Big\} + p_{i1}\Big\{\sum_{v=1}^{V} D_{iv}\log \eta_{v1}^{(t)} + \log \pi^{(t)}\Big\}\right]
\end{aligned}
\tag{2}
$$

where p_{ik} is the posterior probability of a document i being assigned to the k th cluster, $k = \{0, 1\}$, given data and the parameters at t th iteration. If a document has a positive label, $p_{i0} = 0$ and $p_{i1} = 1$.

If a document has no label,

$$p_{i0} \;=\; 1 - p_{i1}$$

$$p_{i1} = \frac{\prod_{v=1}^{V} \eta_{v1}^{D_{iv}} \times \pi}{\prod_{v=1}^{V} \left\{\eta_{v0}^{D_{iv}} \times (1-\pi)\right\} + \prod_{v=1}^{V} \left\{\eta_{v1}^{D_{iv}} \times \pi\right\}} \tag{3}$$

Equation 3 also works as the prediction equation. The predicted class of a document i is k that maximizes this posterior probability.

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.

$$\pi^{(t+1)} = \frac{\alpha_1 - 1 + N^{lp} + \lambda \sum_{i=1}^{N^u} p_{i1}}{\left(\alpha_1 - 1 + N^{lp} + \lambda \sum_{i=1}^{N^u} p_{i1}\right) + \left(\alpha_0 - 1 + N^{ln} + \lambda \sum_{i=1}^{N^u} p_{i0}\right)} \tag{4}$$

The updating equation for η is the following.

$$\begin{aligned} \hat{\eta}_{v0}^{(t+1)} &\propto (\beta_{v0} - 1) + \sum_{i=1}^{N^{ln}} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{i0} D_{iv}, \quad v = 1, \ldots, V \\ \hat{\eta}_{v1}^{(t+1)} &\propto (\beta_{v1} - 1) + \sum_{i=1}^{N^{lp}} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{i1} D_{iv}, \quad v = 1, \ldots, V \end{aligned} \tag{5}$$

C EM algorithm for binary classification with multiple clusters

C.1 Summary

The model outlined above assumes that there are two latent clusters, each linked to the positive and the negative class. However, this assumption can be relaxed to link multiple clusters to the negative class. In the world of mixture models, the simplest setup is to let $K = 2$ since the classification goal is binary, and we can link each latent cluster to the final classification categories. A more general setup is to use $K > 2$ even when a goal is a binary classification. If $K > 2$, but our focus is to uncover the identity of one cluster, we can choose one of the latent clusters to be linked to the "positive" class and let all other latent clusters be linked to the "negative" class (see e.g., Larsen and Rubin 2001 for a similar idea in the realm of record linkage). In other words, we collapse the $K - 1$ latent clusters into one class for the classification purpose. Using $K > 2$ makes sense if the "negative" class consists of multiple sub-categories. For instance, suppose researchers are interested in classifying news articles into political news or not. Then, it is reasonable to assume that the non-political news category consists of multiple sub-categories, such as technology, entertainment, and sports news.

C.2 Model

This section presents a model and inference algorithm when we use more than 2 latent clusters in estimation but the final classification task is binary. In other words, we impose a hierarchy where many latent clusters are collapsed into the negative class. In contrast, the positive class is made out of just one class. The model presented is as follows:

$$\begin{aligned} \pi &\sim Dirichlet(\boldsymbol{\alpha}) \\ Z_i &\overset{i.i.d}{\sim} Categorical(\boldsymbol{\pi}) \\ \eta_{\cdot k} &\overset{i.i.d}{\sim} Dirichlet(\boldsymbol{\beta}_k), \quad k = \{1, \ldots, K\} \\ \mathbf{D}_{i\cdot} | Z_i = k &\overset{i.i.d}{\sim} Multinomial(n_i, \boldsymbol{\eta}_{\cdot k}) \end{aligned} \tag{6}$$

Note that $\boldsymbol{\pi}$ is now a probability vector of length K, and it is drawn from a Dirichlet distribution.

Let k^* be the index of the cluster linked to the positive class. The observed likelihood is

the following.

$$
\begin{aligned}
&p(\boldsymbol{\pi}, \boldsymbol{\eta} | \mathbf{D}, \mathbf{C}^{lp}, \mathbf{C}^{ln}) \\
&\propto p(\boldsymbol{\pi}) p(\boldsymbol{\eta}) p(\mathbf{D}^{lp}, \mathbf{C}^{lp} | \boldsymbol{\pi}, \boldsymbol{\eta}) p(\mathbf{D}^{ln}, \mathbf{C}^{ln} | \boldsymbol{\pi}, \boldsymbol{\eta}) \Big[p(\mathbf{D}^{u} | \boldsymbol{\pi}, \boldsymbol{\eta}) \Big]^{\lambda} \\
&= p(\boldsymbol{\pi}) p(\boldsymbol{\eta}) \times \prod_{i=1}^{N^{lp}} p(\mathbf{D}_i^{lp} | Z_i = k^*, \eta) p(Z_i = k^* | \boldsymbol{\pi}) \\
&\quad \times \prod_{i=1}^{N^{ln}} \sum_{k \neq k^*} \Big\{ p(\mathbf{D}_i^{ln} | Z_i = k, \eta) p(Z_i = k | \boldsymbol{\pi}) \Big\} \times \left[\prod_{i=1}^{N^{u}} \sum_{k=1}^{K} \Big\{ p(\mathbf{D}_i^{u} | Z_i = k, \boldsymbol{\eta}) p(Z_i = k | \boldsymbol{\pi}) \Big\} \right]^{\lambda} \\
&\propto \underbrace{\prod_{k=1}^{K} \left\{ \pi_k^{\alpha_k - 1} \prod_{v=1}^{V} \eta_{vk}^{\beta_{kv} - 1} \right\}}_{\text{prior}} \times \underbrace{\prod_{i=1}^{N^{lp}} \Big\{ \prod_{v=1}^{V} \eta_{vk^*}^{D_{iv}} \times \pi_k \Big\}}_{\text{positive labeled doc. likelihood}} \\
&\quad \times \underbrace{\prod_{i=1}^{N^{ln}} \sum_{k \neq k^*} \Big\{ \prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k \Big\}}_{\text{negative labeled doc. likelihood}} \times \underbrace{\left[\prod_{i=1}^{N^{u}} \sum_{k=1}^{K} \Big\{ \prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k \Big\} \right]^{\lambda}}_{\text{unlabeled doc. likelihood}}
\end{aligned}
\tag{7}
$$

The Q function (the expectation of the complete log likelihood) is

$$
\begin{aligned}
Q &\equiv \mathbb{E}_{\mathbf{Z} | \boldsymbol{\pi}^{(t)}, \boldsymbol{\eta}^{(t)}, D, C} [p(\boldsymbol{\pi}, \boldsymbol{\eta}, \mathbf{Z} | \mathbf{D}, \mathbf{C})] \\
&= \sum_{k=1}^{K} \left[(\alpha_k - 1) \log \pi_k^{(t)} + \sum_{v=1}^{V} \Big\{ (\beta_{kv} - 1) \log \eta_{vk}^{(t)} \Big\} \right] \\
&\quad + \sum_{i=1}^{N^{lp}} \Big\{ \sum_{v=1}^{V} D_{iv} \log \eta_{vk^*}^{(t)} + \log \pi_{k^*}^{(t)} \Big\} + \sum_{i=1}^{N^{ln}} \sum_{k \neq k^*} p_{ik} \Big\{ \sum_{v=1}^{V} D_{iv} \log \eta_{vk}^{(t)} + \log \pi_k^{(t)} \Big\} \\
&\quad + \lambda \left[\sum_{i=1}^{N^{u}} \sum_{k=1}^{K} p_{ik} \Big\{ \sum_{v=1}^{V} D_{iv} \log \eta_{vk}^{(t)} + \log \pi_k^{(t)} \Big\} \right]
\end{aligned}
\tag{8}
$$

The posterior probability of $Z_i = k$, p_{ik}, is

$$
p_{ik} = \frac{\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k}{\sum_{k=1}^{K} \left[\prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k \right]} \tag{9}
$$

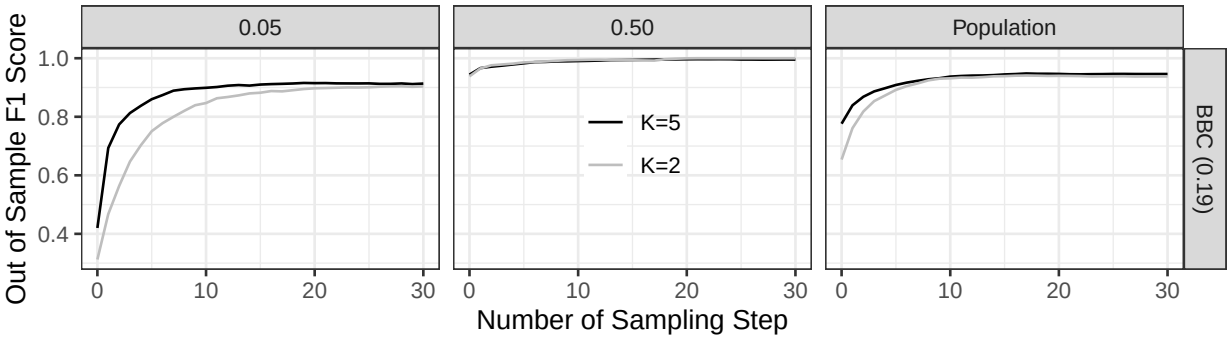

Figure C.1: **Classification Results with 2 and 5 Clusters.**
The darker lines show the results with 5 latent clusters and the lighter lines show 2 latent clusters. The columns correspond to various proportions of positive labels in the corpus. The y-axis indicates the out-of-sample F1 score and the x-axis show the number of sampling steps. Using multiple clusters improves the classification performance when the number of latent clusters matches the data generating process.

M step estimators are The updating equation for π is the following.

$$\hat{\pi_k} \propto \begin{cases} \alpha_k - 1 + \sum_{i=1}^{N^{ln}} p_{ik} + \lambda \sum_{i=1}^{N^u} p_{ik} & \text{if } k \neq k^* \\ \alpha_k - 1 + N^{lp} + \lambda \sum_{i=1}^{N^u} p_{ik^*} & \text{if } k = k^* \end{cases} \tag{10}$$

The updating equation for η is the following.

$$\hat{\eta}_{vk} \propto \begin{cases} (\beta_k - 1) + \sum_{i=1}^{N^{ln}} p_{ik} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{ik} D_{iv} & \text{if } k \neq k^* \\ (\beta_k - 1) + \sum_{i=1}^{N^{lp}} D_{iv} + \lambda \sum_{i=1}^{N^u} p_{ik^*} D_{iv} & \text{if } k = k^* \end{cases} \tag{11}$$

Note that we downweight the information from the unlabeled documents by λ, to utilize more reliable information from labeled documents.

C.3 Results

Figure C.1 shows the results of a model with just two latent clusters vs. a model with 5 latent clusters but only two final classes (positive vs. negative). The darker lines show the results with 5 latent clusters and the lighter lines show the results with 2 latent clusters. Overall, the model with 5 clusters performs better or as well as the model with 2 clusters. The gain from using 5 clusters is the highest when the proportion of positive labels is small and when the size of labeled data is small.

Figure C.2 shows the results when the multiple cluster approach and keyword upweighting approaches are combined.

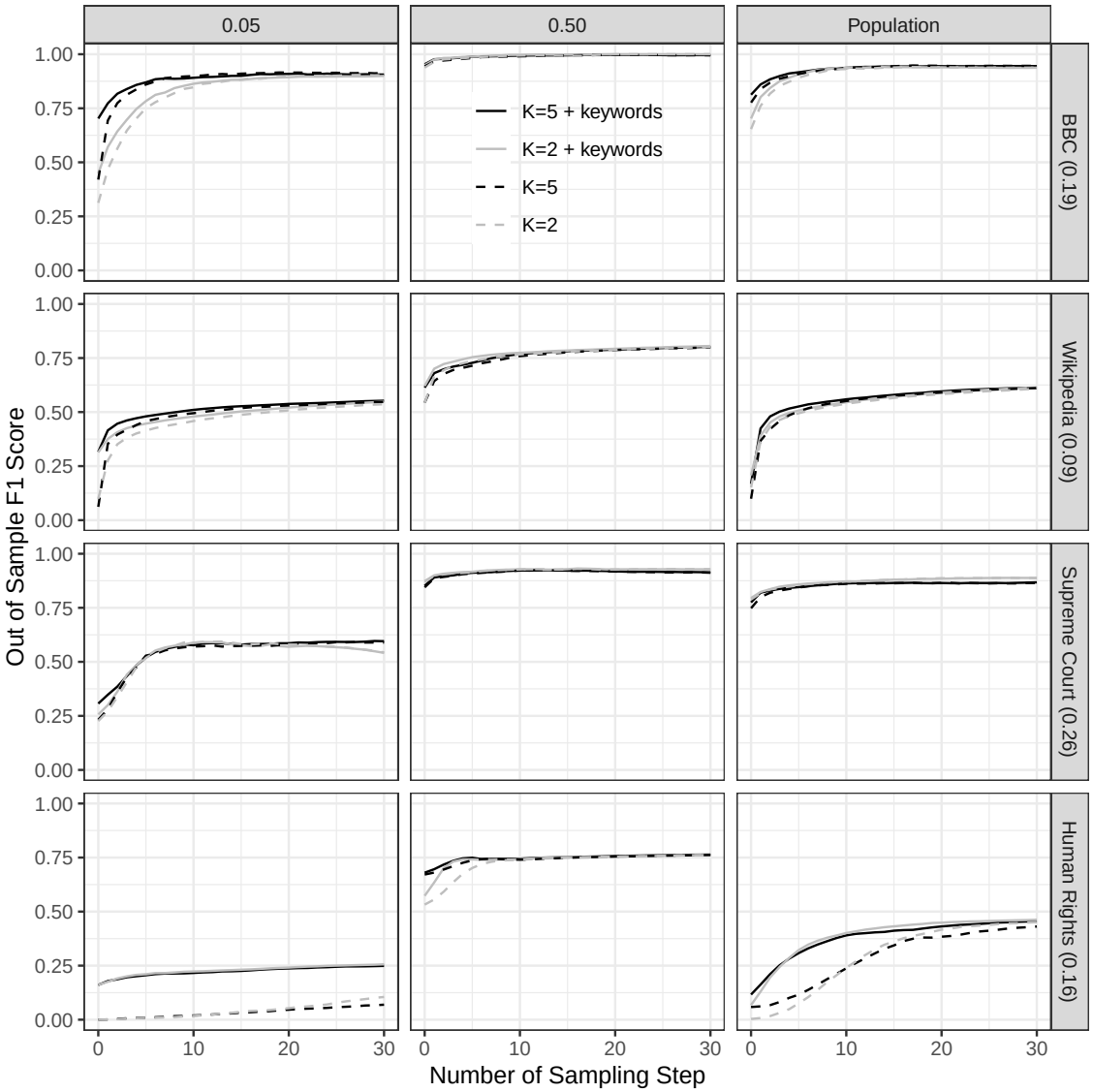

Figure C.2: **Classification Results with Multiple Clusters and Keywords.** The rows correspond to different datasets and the columns correspond to various proportions of positively labeled documents in the corpus. The y-axis indicates the out-of-sample F1 score and the x-axis show the number of sampling steps. The linetype show whether keywords are supplied: the solid lines show the results with keywords and the dashed lines without keywords. The colors show the number of latent clusters in the mixture model: the darker lines show the results with 5 latent clusters and the lighter lines with 2 latent clusters. Using 5 clusters leads to as good or slightly better performance than using 2 clusters. The performance improvement is the largest with the BBC corpus, which consists of 5 news topic categories. Likewise, our mixture models with keywords leads to as good or better performance than the models without keywords. The improvement is the largest with the human rights corpus, where the number of words per document is the smallest.

D Multiclass Classification

D.1 Model

This section presents a model and inference algorithm for multiclass classification. Let K be the number of the clusters and is equal to the number of classes to be classified, with $K \geq 2$. Differently than in SI C, we do not impose any hierarchies and the model is a true multi-class mixture model, where the end goal is to classify documents in $K \geq 2$ classes. In other words, the model presented below is a generalization of the model presented in the main text.

$$
\begin{aligned}
\pi &\sim Dirichlet(\boldsymbol{\alpha}) \\
Z_i &\overset{i.i.d}{\sim} Categorical(\boldsymbol{\pi}) \\
\eta_{\cdot k} &\overset{i.i.d}{\sim} Dirichlet(\boldsymbol{\beta}_k), \quad k = \{1, \ldots, K\} \\
\mathbf{D}_{i\cdot} | Z_i = k &\overset{i.i.d}{\sim} Multinomial(n_i, \boldsymbol{\eta}_{\cdot k})
\end{aligned}
\tag{12}
$$

Note that $\boldsymbol{\pi}$ is now a probability vector of length K, and it is drawn from a Dirichlet distribution.

The observed likelihood is the following.

$$
\begin{aligned}
p(\boldsymbol{\pi}, \boldsymbol{\eta} | \mathbf{D}, \mathbf{C}^l) &\propto p(\boldsymbol{\pi}) p(\boldsymbol{\eta}) p(\mathbf{D}, \mathbf{C} | \boldsymbol{\pi}, \boldsymbol{\eta}) \Big[p(\mathbf{D}^u | \boldsymbol{\pi}, \boldsymbol{\eta}) \Big]^{\lambda} \\
&= p(\boldsymbol{\pi}) p(\boldsymbol{\eta}) \times \prod_{k=1}^{K} \prod_{i=1}^{N^k} p(\mathbf{D}_i^l | Z_i = k, \eta) p(Z_i = k | \boldsymbol{\pi}) \\
&\quad \times \left[\prod_{i=1}^{N^u} \sum_{k=1}^{K} \Big\{ p(\mathbf{D}_i^u | Z_i = k, \boldsymbol{\eta}) p(Z_i = k | \boldsymbol{\pi}) \Big\} \right]^{\lambda} \\
&\propto \underbrace{\prod_{k=1}^{K} \left\{ \pi_k^{\alpha_k - 1} \prod_{v=1}^{V} \eta_{vk}^{\beta_{kv} - 1} \right\}}_{\text{prior}} \times \underbrace{\prod_{k=1}^{K} \prod_{i=1}^{N^k} \Big\{ \prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k \Big\}}_{\text{labeled doc. likelihood}} \times \underbrace{\left[\prod_{i=1}^{N^u} \sum_{k=1}^{K} \Big\{ \prod_{v=1}^{V} \eta_{vk}^{D_{iv}} \times \pi_k \Big\} \right]^{\lambda}}_{\text{unlabeled doc. likelihood}}
\end{aligned}
\tag{13}
$$

The Q function (the expectation of the complete log-likelihood) is

$$
\begin{aligned}
Q &\equiv \mathbb{E}_{\mathbf{Z}|\boldsymbol{\pi}^{(t)},\boldsymbol{\eta}^{(t)},D,C}[p(\boldsymbol{\pi},\boldsymbol{\eta},\mathbf{Z}|\mathbf{D},\mathbf{C})] \\
&= \sum_{k=1}^{K}\left[(\alpha_k-1)\log\pi_k^{(t)} + \sum_{v=1}^{V}\left\{(\beta_{kv}-1)\log\eta_{vk}^{(t)}\right\}\right] \\
&\quad + \sum_{k=1}^{K}\sum_{i=1}^{N^k}\left\{\sum_{v=1}^{V} D_{iv}\log\eta_{vk}^{(t)} + \log\pi_k^{(t)}\right\} \\
&\quad + \lambda\left[\sum_{i=1}^{N^u}\sum_{k=1}^{K} p_{ik}\left\{\sum_{v=1}^{V} D_{iv}\log\eta_{vk}^{(t)} + \log\pi_k^{(t)}\right\}\right]
\end{aligned}
\tag{14}
$$

The posterior probability of $Z_i = k$, p_{ik}, is

$$
p_{ik} = \frac{\prod_{v=1}^{V}\eta_{vk}^{D_{iv}} \times \pi_k}{\sum_{k=1}^{K}\left[\prod_{v=1}^{V}\eta_{vk}^{D_{iv}} \times \pi_k\right]} \tag{15}
$$

M step estimators are The updating equation for π is the following.

$$
\hat{\pi_k} \propto \alpha_k - 1 + N^k + \lambda\sum_{i=1}^{N^u} p_{ik} \tag{16}
$$

The updating equation for η is the following.

$$
\hat{\eta}_{vk} \propto (\beta_k - 1) + \sum_{i=1}^{N^k} D_{iv} + \lambda\sum_{i=1}^{N^u} p_{ik}D_{iv} \tag{17}
$$

Note that we downweight the information from the unlabeled documents by λ, to utilize more reliable information from labeled documents.

D.2 Results

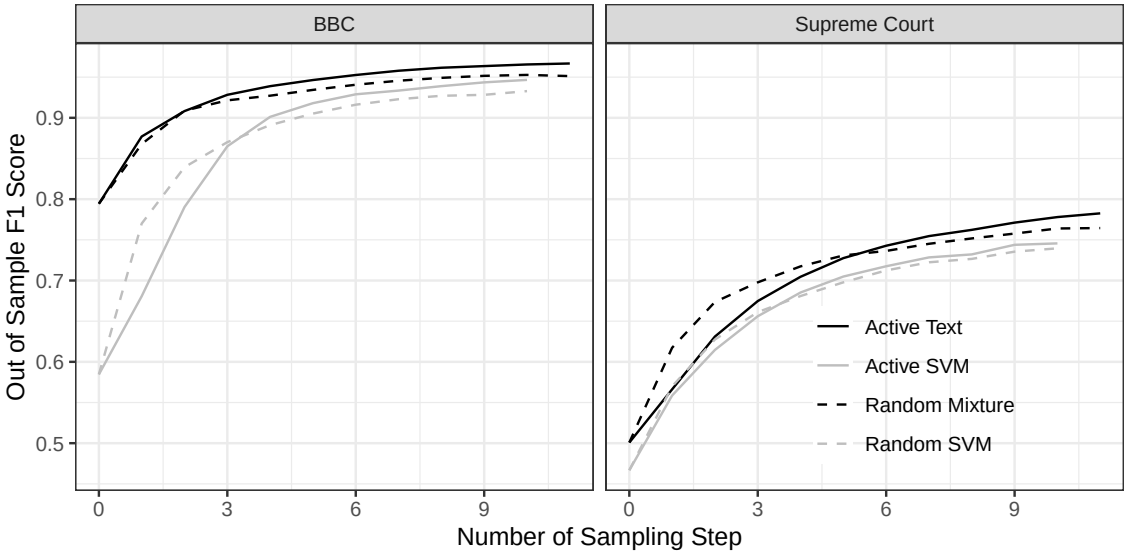

Figure D.1: **Multiclass Classification Results.**
The darker lines show the results with *activeText* and the lighter lines show the results with SVM. The solid lines use active sampling to decide the next set of documents to be labeled, and the dashed lines use random (passive) sampling. The y-axis indicates the out-of-sample F1 score and the x-axis show the number of sampling steps. The left column shows the results on BBC corpus, where the target classes are "Politics," "Entertainment," "Business," "Sports," and "Technology." "Politics" class has 5% of the total dataset, and the rest 95% is evenly split across the rest of classes. The right column shows the results on the Supreme Court corpus, where the target classes are "Criminal Procedure" (32.4% of the corpus), "Civil Rights" (21.4%), "Economic Activity" (22.2%), "Judicial Power" (15.4%), "First Amendment (8.6%)." In our model, we set the number of latent clusters to be the same as the classification categories and linked each latent cluster to one classification category. *activeText* performs the best across the four specifications on both corpora.

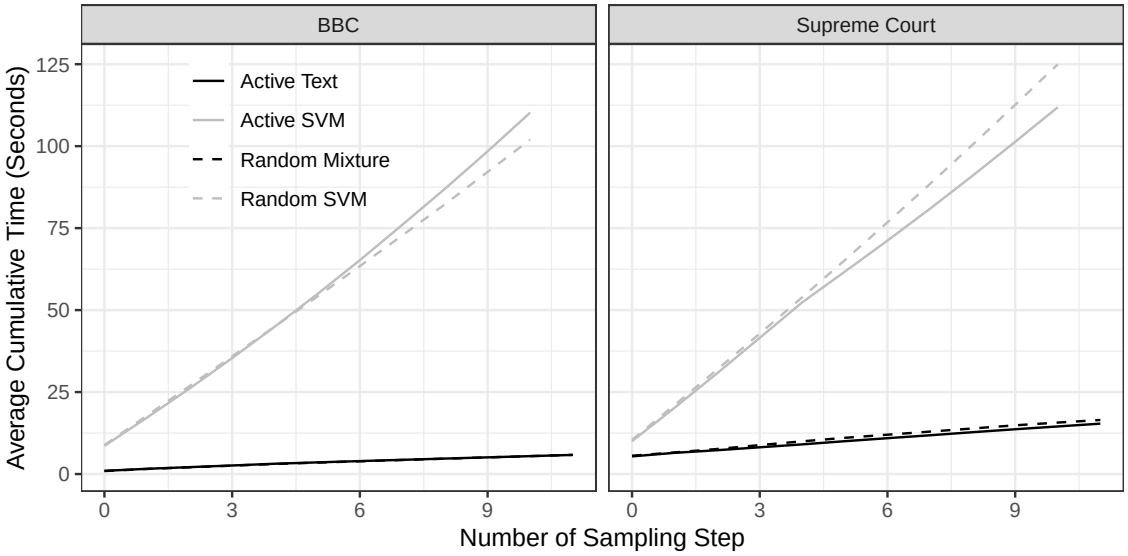

Figure D.2: **Time comparison of Multiclass Classification Results.**
The darker lines show the results with *activeText* and the lighter lines show the results with SVM. The solid lines use active sampling to decide the next set of documents to be labeled, and the dashed lines use random (passive) sampling. The y-axis indicates the average cumulative computational time and the x-axis shows the number of sampling steps. The left column shows the results on BBC corpus, and the right column shows the results on the Supreme Court corpus. *activeText* is much faster than SVM in multiclass classification. This is because multiclass classification with SVM requires fitting the model repeatedly at least the same time as the number of target classes. By contrast, *activeText* requires to fit only once regardless of the number of target classes.

E Model Specifications and Description of the Datasets in the Validation Performance

We explain our decisions regarding pre-processing steps, model evaluation, and model specifications, followed by a detailed discussion of the results for each dataset.

E.1 Pre-processing

We employ the same pre-processing step for each of the four 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.

To generate dataset with the proportion of positive class p (e.g. 5% or 50%), we randomly sample documents from the original dataset so that it achieves the proportion of the positive class p. Suppose the number of documents in the original dataset is N with N_{pos} and N_{neg} the number of positive and negative documents, respectively. We compute $M_{pos} = \text{floor}(Np)$ and $M_{neg} = N - M_{pos}$ as the ideal numbers of positive and negative documents. While $M_{pos} > N_{pos}$ or $M_{neg} > N_{neg}$, we decrement M_{pos} and M_{neg} keeping the positive proportion to p. With $M_{pos} < N_{pos}$ and $M_{neg} < N_{neg}$, we sample M_{pos} positive documents and M_{neg} negative documents from the original dataset. Finally, combine the sampled positive and negative documents to obtain the final dataset.

E.2 Datasets

BBC News The BBC News Dataset is a collection of 2,225 documents from 2004 to 2005 available at the BBC news website (Greene and Cunningham, 2006). This dataset is 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.

Wikipedia Toxic Comments 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,[9] 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

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

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

of other users.[10] The complete dataset is comprised of roughly 560,000 documents, roughly 10 percent of which are labeled as toxic.

Supreme Court Cases The Supreme Court Rulings dataset is a collection of the text of 2000 US Supreme Court rulings between 1946 and 2012. We use the majority opinion of each case and the text was obtained through Caselaw Access Project.[11] For the classification label, we use the categories created by the Supreme Court Database.[12] The classification exercise here is to correctly identify rulings that are categorized as 'criminal procedure', which is the largest category in the corpus (26% of all rulings).

Human Rights Allegation Human Rights Allegation dataset contains more than 2 million sentences of human rights reports in 196 countries between 1996 and 2016, produced by Amnesty International, Human Rights Watch and the US State Department (Cordell et al., 2021). The classification goal is to identify sentences with physical integrity rights allegation (16% of all reports). Example violations of physical integrity rights include torture, extrajudicial killing, and arbitrary arrest and imprisonment.

[10]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.

[11]`https://case.law`

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

F Additional Results on Classification Performance

To complement the results presented in Figure 1 in the main text, Table F.1 presents the results (across datasets) of fitting our model at the initial (iteration 0) and last active step (iteration 30). It is clear from the table that the improvements *activeText* brings in terms of the F1-score, precision, and recall. Furthermore, after labeling 600 documents (20 per iteration), uncertainty sampling outperforms random sampling across evaluation metrics, which empirically validates the promise of active learning in terms of text classification.

Table F.1: **Classification Performance: Uncertainty vs Random Sampling with** $\lambda = 0.001$

Dataset	Active Step	Uncertainty Sampling			Random Sampling		
		Precision	Recall	F1-score	Precision	Recall	F1-score
`Wikipedia`	0	0.71	0.13	0.22	0.71	0.13	0.22
	30	0.71	0.54	0.61	0.45	0.56	0.50
`BBC`	0	0.33	0.86	0.48	0.33	0.86	0.48
	30	0.92	0.96	0.94	0.92	0.94	0.93
`Supreme Court`	0	0.46	0.98	0.63	0.46	0.98	0.63
	30	0.85	0.91	0.88	0.75	0.96	0.84
`Human Rights`	0	0.61	0.01	0.02	0.61	0.01	0.02
	30	0.53	0.42	0.47	0.46	0.44	0.45

Similarly, and as noted in the main text, our results appear to be not too sensitive to the selection of the weighting parameter λ, provided that its value remains small. Figures F.1 confirms this finding. After 30 active steps, the performance of *activeText* is better in terms of F1-score when $\lambda = 0.001$ if compared to $\lambda = 0.01$

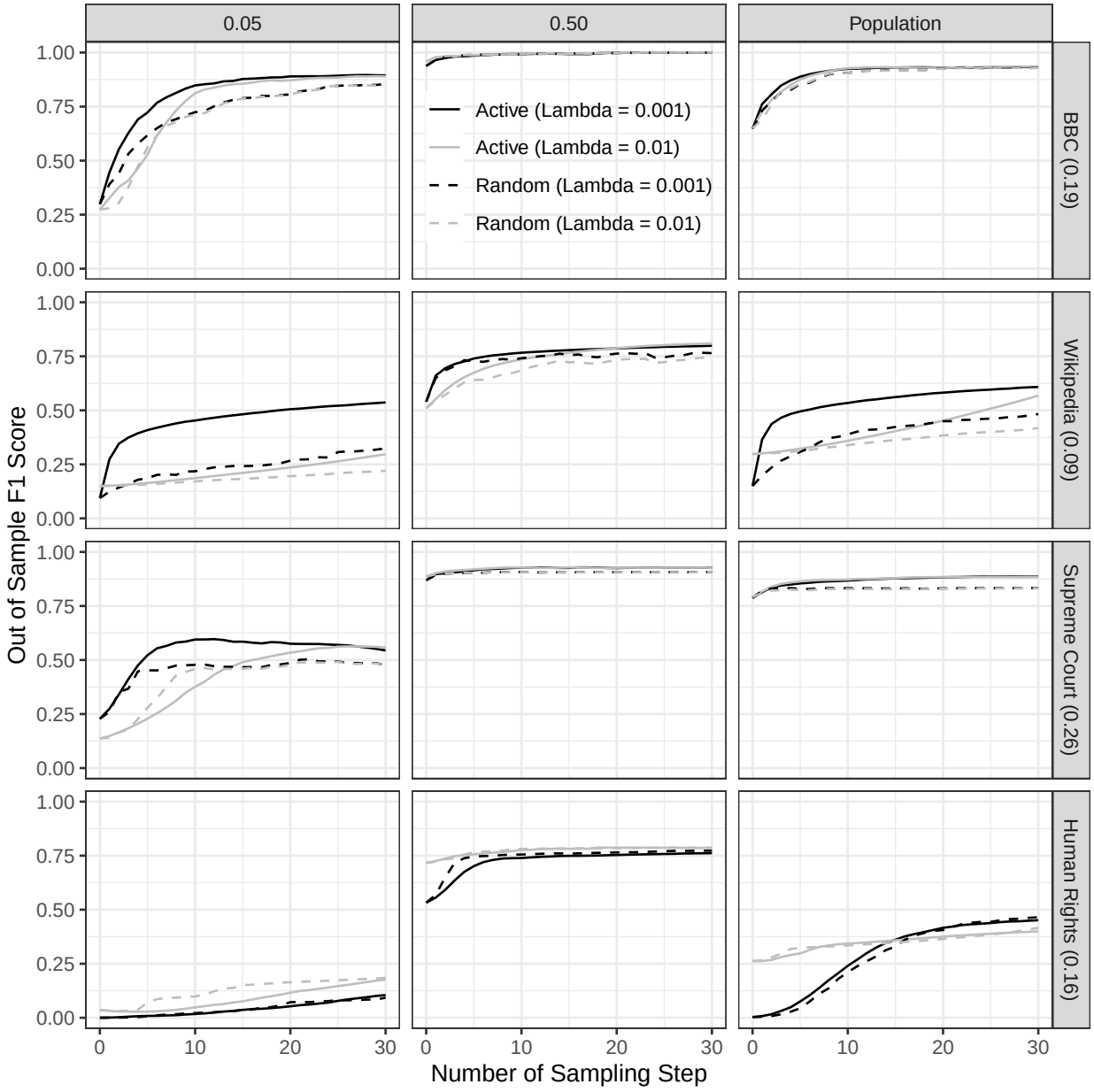

Figure F.1: **Classification Results with 2 Clusters and $\lambda = 0.01$ vs $\lambda = 0.001$.** The darker lines show the results with $\lambda = 0.001$ and the lighter lines show $\lambda = 0.01$. The columns correspond to various proportion of positive labels in the corpus. The y-axis indicates the out-of-sample F1 score and the x-axis show the number of sampling steps. The smaller the value of λ the better the performance of our model.

G Main Results with SVM Comparisons

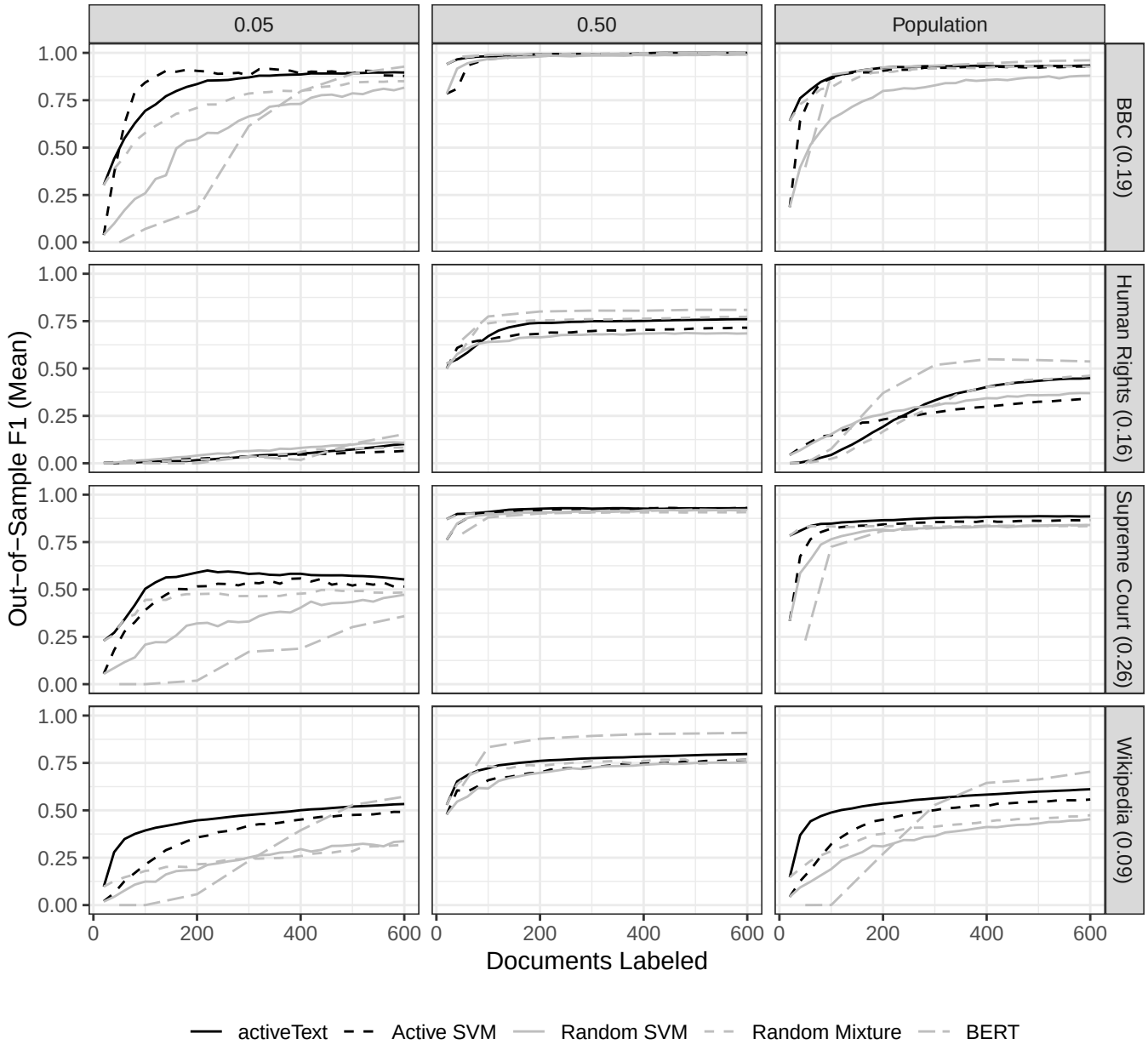

Figure G.1: **Replication of F1 performance from Figures 2 with random and active SVM**

H Visual Demonstration of Updating the Word-Class Matrix

Figure H.1 illustrates how the word-class matrix $\boldsymbol{\eta}$ is updated with and without keywords across iterations. In the generative model for *activeText* , two factors determine the importance of words for the classification. The first is the ratio of the probability that a word appears in a document with a positive vs. negative class, and the second is the word frequency. As these two values increase, the importance of a word increases. In Figure H.1, the former is presented on the x-axis and the latter on the y-axis. Specifically, the x-axis shows the log of η_{v1}/η_{v0}, where η_{v1} corresponds the probability of observing the word v in a document with a positive class and η_{v0} for a document with a negative class, and the y-axis is the log of word frequency. This means that words towards the upper right corner of the figure are more important for the classification. Therefore, this visualization allows users to interpret which words are more important to the classification.

This visualization also illustrates how the keywords are used to update the word-class matrix. By shifting the value of $\boldsymbol{\eta}$ of keywords, which typically appear on the right side of the figures, we can accelerate the estimation of $\boldsymbol{\eta}$ and improve the classification performance. A subset of the keywords supplied is labeled and highlighted by black dots.

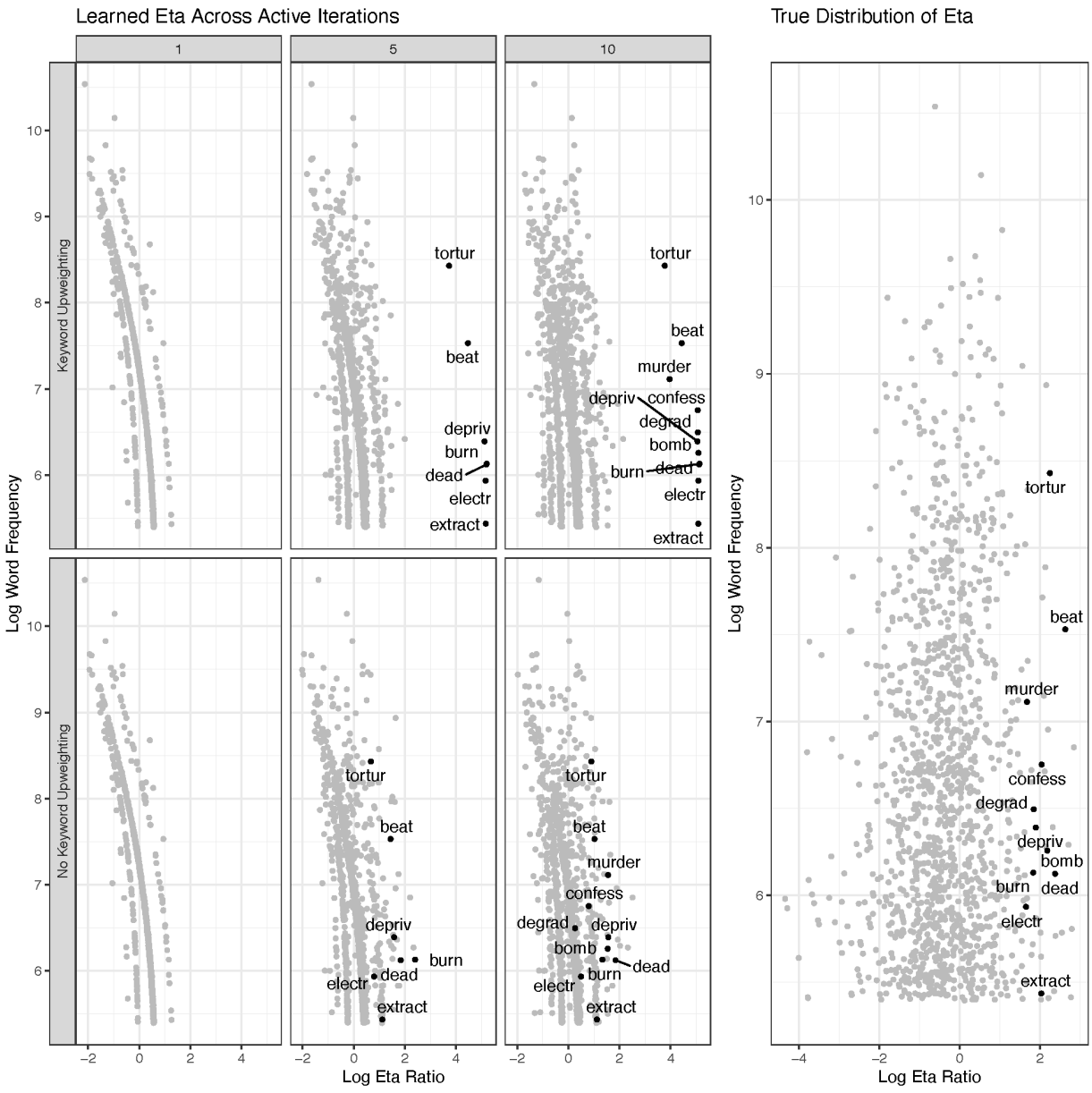

Figure H.1: **Update of the Word-class Matrix (η) with and without Keywords**

I Classification Performance with Mislabels

I.1 Mislabeled Keywords

The rows correspond to different datasets and the columns correspond to various values of γ, which controls the degree of keyword upweighting. The y-axis indicates the out-of-sample F1 score and the x-axis shows the number of sampling steps. At each sampling step, 20 documents are labeled. We use $\lambda = 0.001$ to downweight information from unlabeled documents. The lines correspond to different levels of mislabels at the keyword labeling. At each iteration, 10 candidate keywords are proposed, and a hypothetical oracle decides if they are indeed keywords or not. 'True' keywords are defined in the same way as in Section Benefits of Keyword Upweighting. In other words, a candidate keyword v for the positive class is a 'true' keyword, if the value of $\eta_{v,k}/\eta_{v,k'}$ is above 90% quantile, where k is the positive class and k' is the negative class, and this $\boldsymbol{\eta}$ is what we obtain by training the model with the full labels. The same goes for the negative class. When the probability of mislabeling keywords is p%, an oracle makes a mistake in the labeling with probability p. Specifically, if a candidate keyword v is a 'true' keyword, the oracle would not label v as a keyword with probability p. Likewise, if a candidate keyword v is not a 'true' keyword, they would label v as a keyword.

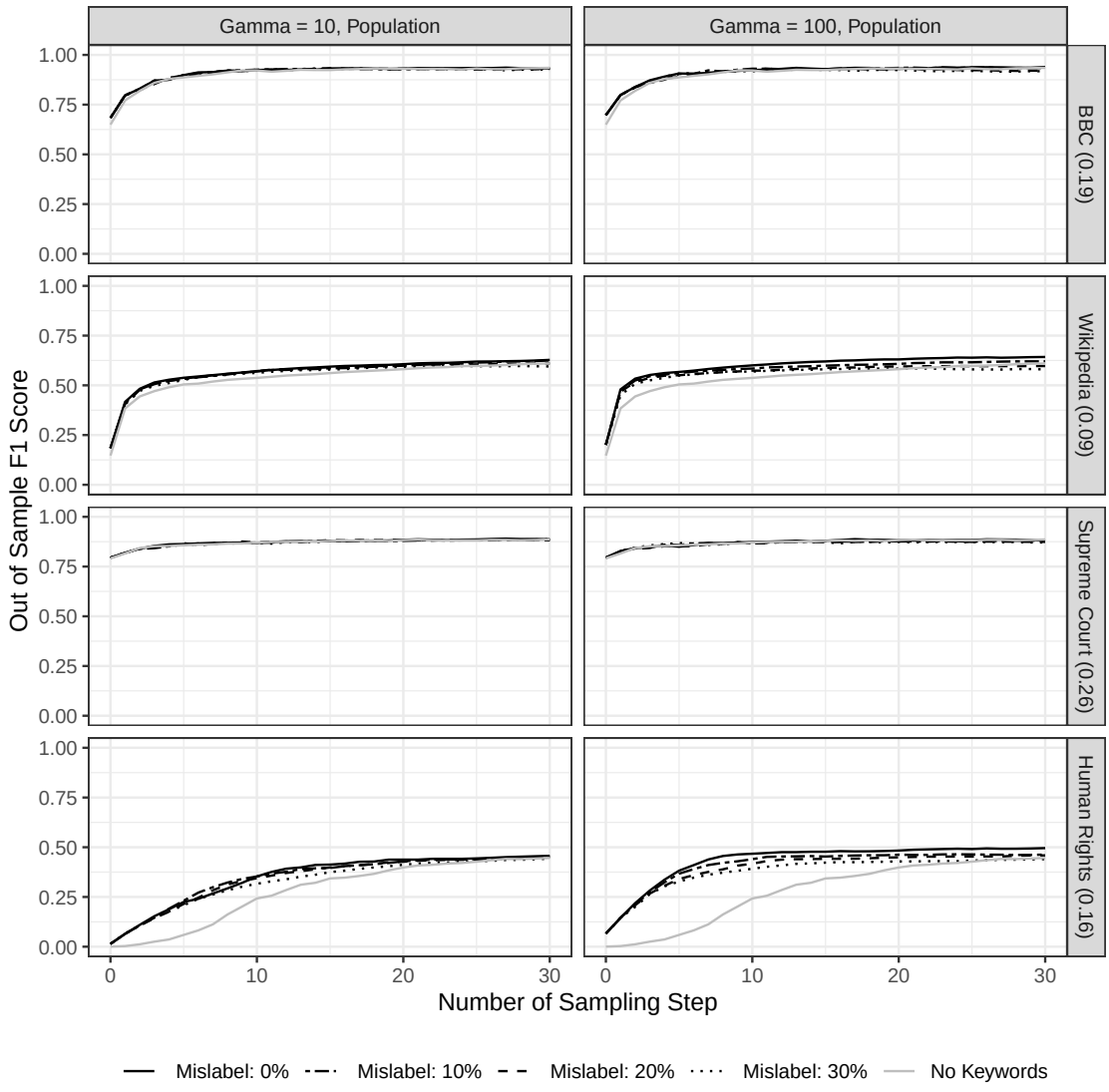

Figure I.1: **Classification Results with Mislabels in Active Keywords**

I.2 Mislabeled Documents

In this section, we present results about the effect of 'honest' (random) mislabeling of documents on the mapping of documents to classes. As Figure I.2 shows, as the proportion of mislabels increases, the classification performance of *activeText* decreases.

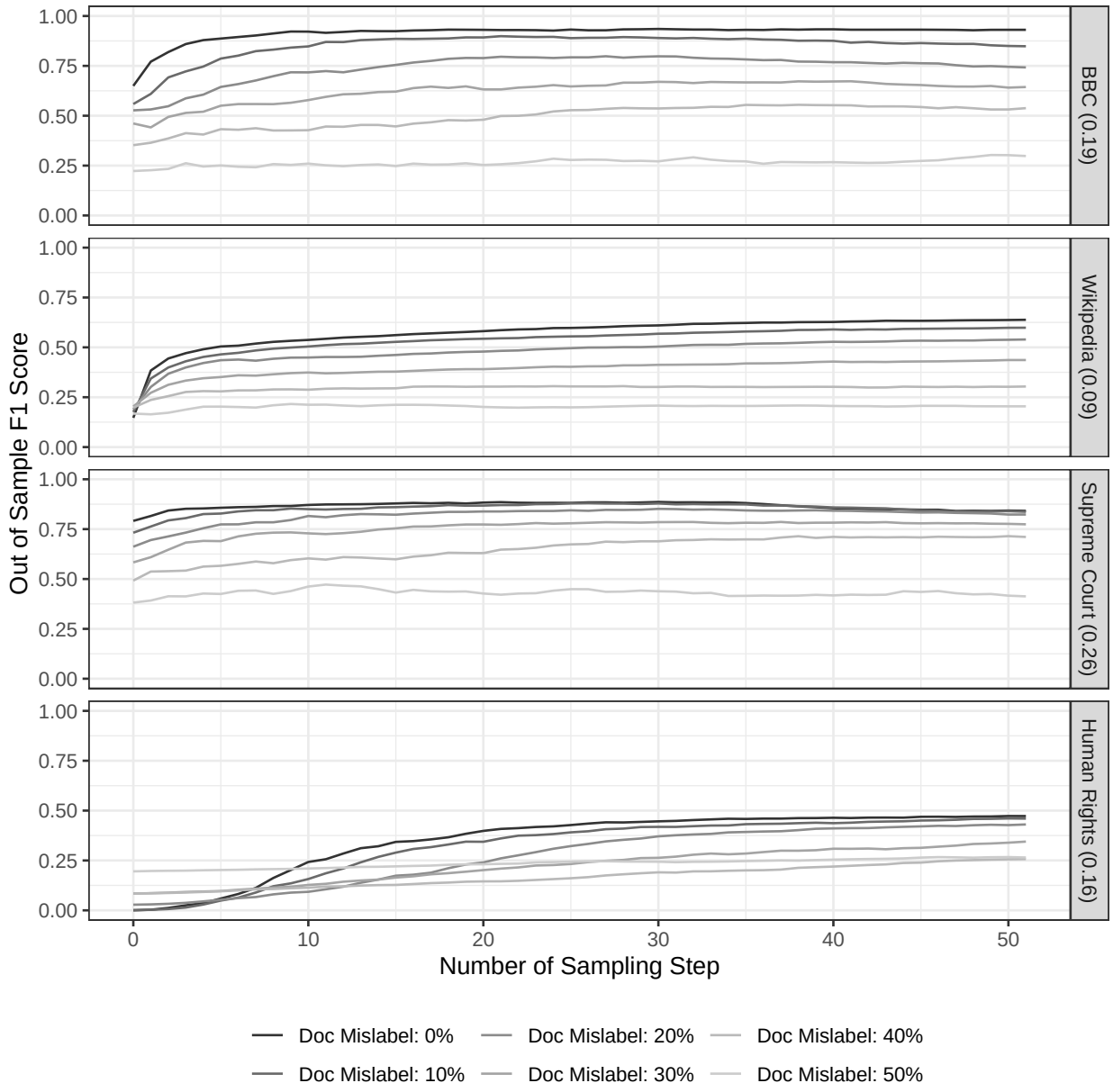

Figure I.2: **Classification Results with Mislabels in Active Document Labeling** The rows correspond to different datasets. The y-axis indicates the out-of-sample F1 score and the x-axis shows the number of sampling steps. 20 documents are labeled at each sampling step. The colors correspond to different levels of mislabels in the labeling of documents. We find that as the proportion of mislabels increases, the classification performance of *activeText* decreases.

J Comparison of the predictions between *activeText* and xgboost predictions for the Gohdes (2020) data

Table J.1 shows the confusion matrix between the prediction based on *activeText* and the prediction by xgboost used in the original paper. Most observations fall in the diagonal cells of the matrix, and the correlation between the two predictions is quite high (0.93). One difference is that *activeText* classifies more documents to target killings compared to the original predictions. Note that either prediction claims the ground truth. Both are the results of different classifiers. Figure J.1 confirms these findings when we focus on the proportion of biweekly government targeted killings constructed via *activeText* when compared to the same construct in Gohdes (2020).

		Original		
		untargeted	targeted	non-government
activeText	untargeted	50327	411	135
	targeted	1630	10044	31
	non-government	382	34	2280

Table J.1: Confusion matrix between *activeText* and xgboost predictions

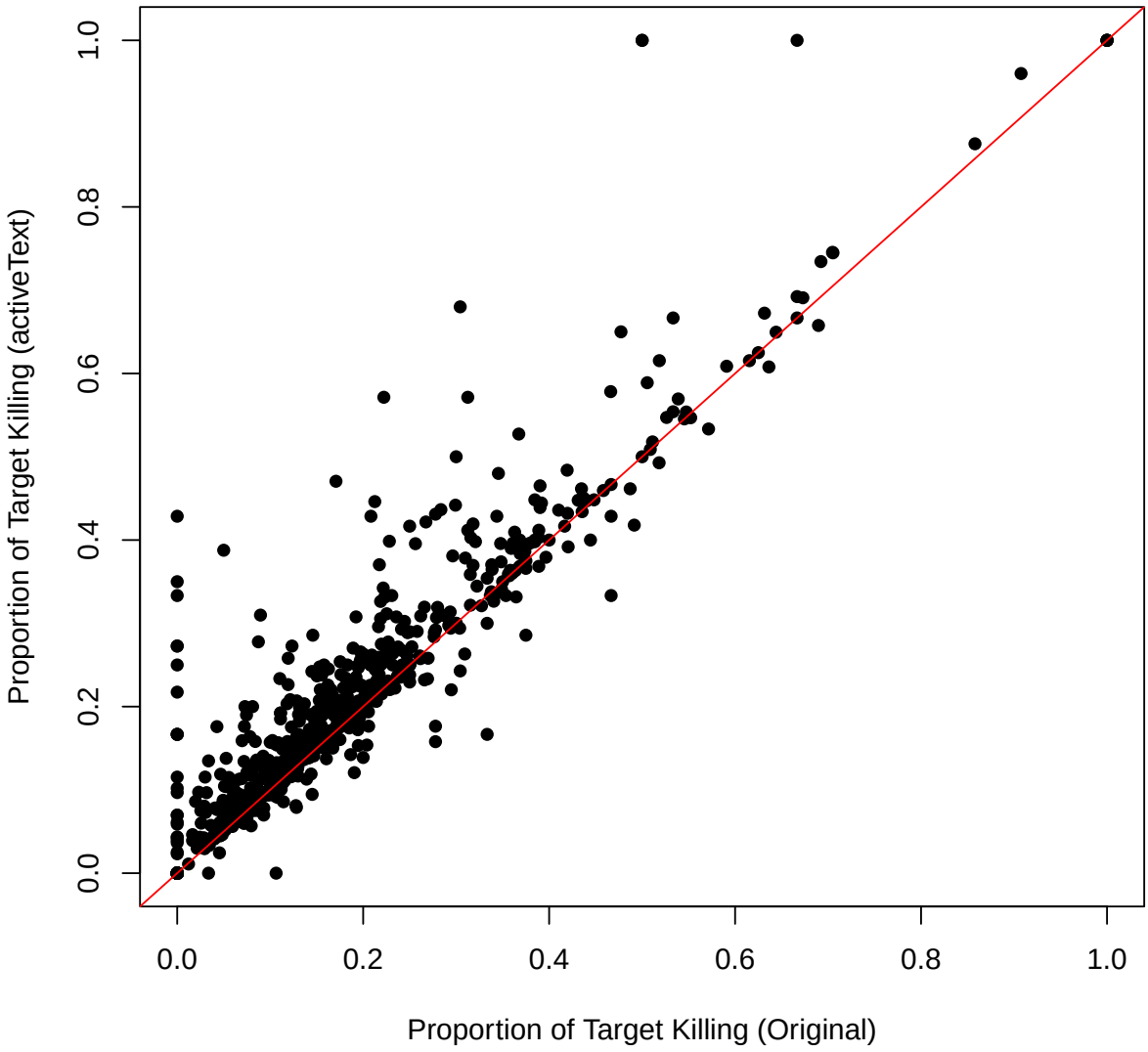

Figure J.1: **Scatter plot of the dependent variable between the one constructed by *activeText* vs. the original**

The author performs a binomial logit regression where the dependent variable is the ratio of the number of targeted killings to the total number of government killings. We compare the dependent variable used in the original paper vs. the one we constructed using *activeText* . The 45-degree line (in red) corresponds to equality between measures. We can see that most observations lie around the 45-degree line while there are some values in the upper triangle. This suggests that *activeText* yields a similar dependent variable to the original one, while there may be some overestimations of the proportion of target killing with *activeText* .

K Regression Table in Gohdes (2020)

Table K.1 is the original regression table reported in Gohdes (2020) while Table K.2 is a replication of the original table using *activeText* . In both tables, the coefficients on the Internet access variable are positive and statistically significant, which match the author's substantive conclusion. One may wonder why the absolute values of the coefficients on the IS and Internet is larger in Table K.2. However, we believe that this is because the number of observations in the IS control is small (51) and there is almost no variation of the Internet access variable within the observations with IS control, as shown in Figure K.1.

	I	II	III	IV	V	VI	VII
Intercept	−2.340***	−2.500***	−0.899*	−0.410	−0.019	−1.308	−3.013**
	(0.205)	(0.267)	(0.403)	(0.521)	(0.357)	(1.057)	(1.103)
Internet access (3G)	0.224*	0.231*	0.200*	0.205*	0.265*	0.313**	0.909***
	(0.095)	(0.094)	(0.085)	(0.087)	(0.113)	(0.116)	(0.124)
% Govt control							0.016***
							(0.004)
Internet (3G) * % Govt control							−0.014***
							(0.001)
Govt control	0.774*	0.803**	1.167***	1.180***	0.080	0.856**	0.811***
	(0.332)	(0.272)	(0.284)	(0.288)	(0.344)	(0.313)	(0.237)
IS control	2.027***	1.644***	1.045*	−0.324	0.432	0.787	−0.663**
	(0.435)	(0.462)	(0.421)	(0.209)	(0.414)	(0.418)	(0.221)
Kurd control	0.386	−0.243	−0.506	−1.331	−0.402	0.033	−0.616
	(0.594)	(0.843)	(0.760)	(1.134)	(0.745)	(0.802)	(0.432)
Opp control	1.160***	1.252***	0.727*	0.759*	−0.700*	−0.281	−0.176
	(0.298)	(0.317)	(0.293)	(0.296)	(0.283)	(0.342)	(0.164)
Internet (3G) * Govt control	−0.163	−0.182	−0.327**	−0.324**	−0.104	−0.358**	
	(0.132)	(0.117)	(0.119)	(0.122)	(0.133)	(0.120)	
Internet (3G) * IS control	−1.798***	−1.525***	−1.377***		−1.391***	−1.336***	
	(0.220)	(0.281)	(0.251)		(0.264)	(0.261)	
Internet (3G) * Kurd control	−0.133	0.336	0.093	0.895	−0.052	−0.202	
	(0.444)	(0.649)	(0.569)	(0.936)	(0.553)	(0.527)	
Internet (3G) * Opp. control	−0.605***	−0.722***	−0.511**	−0.533***	0.316*	0.286	
	(0.159)	(0.173)	(0.157)	(0.158)	(0.151)	(0.186)	
# Killings (log)			−0.273***	−0.271***	−0.354***	−0.412***	−0.584***
			(0.054)	(0.055)	(0.051)	(0.072)	(0.074)
Govt gains				0.643			
				(0.385)			
Govt losses				0.632			
				(0.413)			
Christian					0.068	0.345**	0.398***
					(0.111)	(0.116)	(0.110)
Alawi					1.479**	−1.167***	−0.812***
					(0.522)	(0.177)	(0.176)
Druze					−0.634***	−0.302	0.135
					(0.191)	(0.191)	(0.190)
Kurd					−0.659***	−0.542*	−0.580**
					(0.194)	(0.237)	(0.212)
Internet (3G) * Alawi					−0.909***		
					(0.163)		
Pop (log)						0.196	0.408**
						(0.149)	(0.150)
Unempl. (%)						−0.016	−0.002
						(0.012)	(0.012)
AIC	11956.847	9993.704	9665.749	9495.591	7671.979	7873.915	7327.796
BIC	12001.524	10239.427	9915.941	9744.552	7944.509	8150.913	7595.858
Log Likelihood	−5968.424	−4941.852	−4776.875	−4691.796	−3774.990	−3874.958	−3603.898
Deviance	9519.651	7466.508	7136.554	7026.891	5132.784	5332.720	4790.601
Num. obs.	640	640	640	626	640	640	640

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$. Reference category: Contested control. Governorate-clustered SEs.

Table K.1: Table 1 in Gohdes 2020: Original table

	I	II	III	IV	V	VI	VII
Intercept	-2.196^{***}	-2.428^{***}	-0.795^{*}	-0.351	-0.037	-1.141	-2.695^{*}
	(0.197)	(0.242)	(0.390)	(0.490)	(0.348)	(1.229)	(1.227)
Internet access (3G)	0.277^{**}	0.282^{***}	0.242^{**}	0.250^{**}	0.342^{***}	0.369^{***}	0.853^{***}
	(0.091)	(0.081)	(0.075)	(0.077)	(0.103)	(0.107)	(0.118)
% Govt control							0.015^{***}
							(0.004)
Internet (3G) * % Govt control							-0.013^{***}
							(0.001)
Govt control	0.625^{*}	0.672^{**}	1.048^{***}	1.058^{***}	0.151	0.843^{**}	0.559^{*}
	(0.319)	(0.255)	(0.269)	(0.273)	(0.358)	(0.300)	(0.249)
IS control	15.157^{***}	15.688^{***}	15.072^{***}	-0.275	14.551^{***}	14.877^{***}	-0.600^{**}
	(1.123)	(1.148)	(1.136)	(0.200)	(1.132)	(1.134)	(0.209)
Kurd control	0.795	0.099	-0.227	-0.440	-0.157	0.334	-0.369
	(0.516)	(0.729)	(0.671)	(1.119)	(0.677)	(0.744)	(0.405)
Opp control	0.978^{***}	1.134^{***}	0.594^{*}	0.634^{*}	-0.606^{*}	-0.197	-0.278
	(0.294)	(0.304)	(0.284)	(0.289)	(0.270)	(0.322)	(0.155)
Internet (3G) * Govt control	-0.169	-0.190	-0.334^{**}	-0.335^{**}	-0.183	-0.408^{***}	
	(0.126)	(0.103)	(0.108)	(0.111)	(0.131)	(0.111)	
Internet (3G) * IS control	-14.829^{***}	-15.506^{***}	-15.351^{***}		-15.392^{***}	-15.330^{***}	
	(1.080)	(1.096)	(1.090)		(1.091)	(1.091)	
Internet (3G) * Kurd control	-0.400	0.138	-0.080	0.134	-0.240	-0.366	
	(0.324)	(0.514)	(0.463)	(0.940)	(0.473)	(0.460)	
Internet (3G) * Opp. control	-0.542^{***}	-0.688^{***}	-0.468^{**}	-0.497^{**}	0.181	0.149	
	(0.159)	(0.164)	(0.150)	(0.152)	(0.145)	(0.176)	
# Killings (log)			-0.278^{***}	-0.274^{***}	-0.356^{***}	-0.415^{***}	-0.567^{***}
			(0.053)	(0.054)	(0.051)	(0.071)	(0.073)
Govt gains				0.512			
				(0.349)			
Govt losses				0.730^{*}			
				(0.334)			
Christian					0.092	0.352^{**}	0.369^{***}
					(0.115)	(0.113)	(0.105)
Alawi					1.329^{*}	-0.928^{***}	-0.585^{***}
					(0.528)	(0.167)	(0.168)
Druze					-0.628^{**}	-0.310	0.063
					(0.196)	(0.197)	(0.209)
Kurd					-0.565^{**}	-0.502^{*}	-0.615^{**}
					(0.204)	(0.227)	(0.207)
Internet (3G) * Alawi					-0.782^{***}		
					(0.164)		
Pop (log)						0.185	0.391^{*}
						(0.167)	(0.168)
Unempl. (%)						-0.019	-0.007
						(0.012)	(0.012)
AIC	12050.644	10116.531	9739.975	9570.556	8038.596	8197.433	7735.527
BIC	12095.321	10362.255	9990.166	9819.517	8311.125	8474.431	8003.589
Log Likelihood	-6015.322	-5003.266	-4813.988	-4729.278	-3958.298	-4036.717	-3807.763
Deviance	9500.059	7475.946	7097.391	6986.658	5386.011	5542.849	5084.942
Num. obs.	640	640	640	626	640	640	640

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$. Reference category: Contested control. Governorate-clustered SEs.

Table K.2: Table 1 in Gohdes 2020: Reanalysis with *activeText*

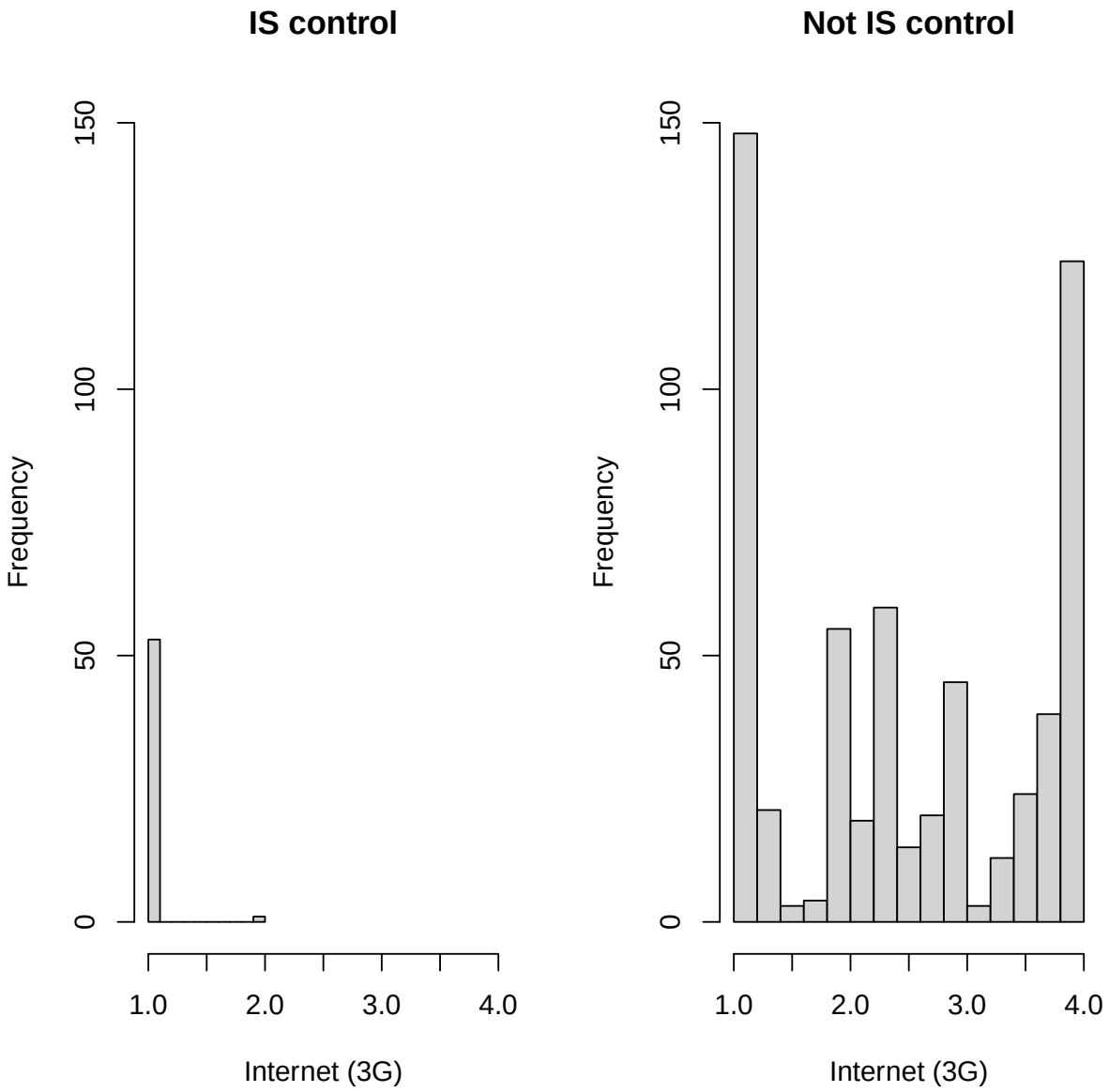

Figure K.1: **Histogram of the Internet (3G) variable by the IS control in the original data**
The left histogram is the distribution of the Internet (3G) variable for the observation under IS control, and the right one is not under IS control. The number of observations with IS control is only 51 out of the total observation of 640. In addition, among those with IS control, all observations except one takes the same value for the Internet access variable. This suggests that the regression coefficient on the interaction of IS control and Internet access can be highly unstable.

L Effect of Labeling More Sentences for the Park et al. (2020) Reanalysis

In this section, we present additional results mentioned in the main text about our reanalysis of Park et al. (2020).

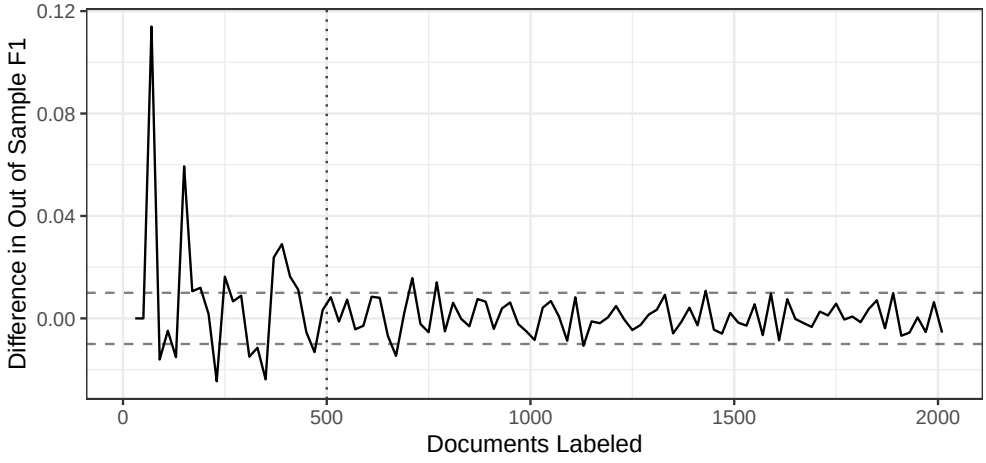

Figure L.1: **Using the Difference in Out of Sample F1 Score to Decide a Stopping Point.**

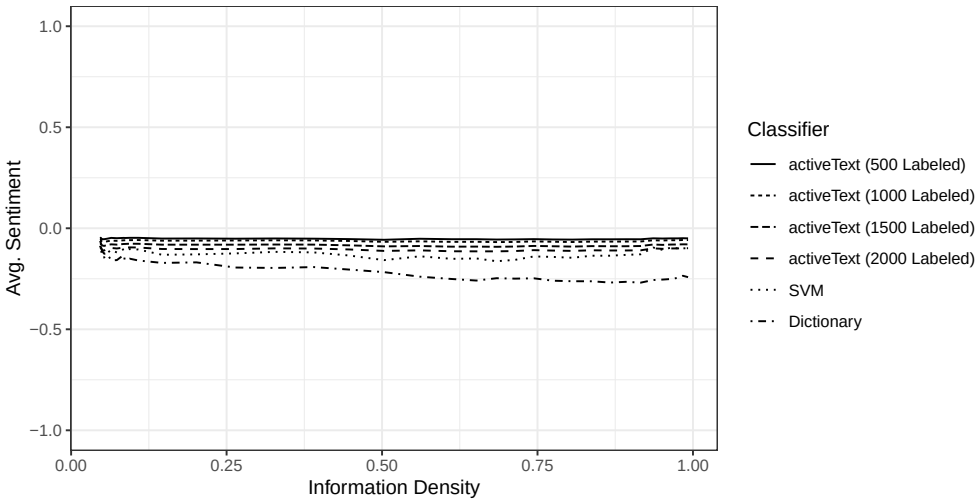

Figure L.2: **Replication of Figure 1 in Park et al. (2020): The Relationship Between Information Density and Average Sentiment Score Across Different Settings for the Total Number of Labeled Documents.**

M Simulation Studies

We conduct a comprehensive set of simulation studies to evaluate the classification performance of our active learning approach. Specifically, across simulations, we examine the classification performance of the proposed method in terms of accuracy, precision, recall, and F1 score. To accomplish this, we systematically vary the structure of the simulated data by manipulating settings such as the corpus size (i.e., the number of documents), vocabulary size, average document length (measured in words), classification difficulty, and the proportion of documents in the positive class within the corpus.

M.1 Simulation Setup

The process of generating simulated data is based on the mixture model that we have defined above (See Section The Method). This allows us to systematically vary many dimensions of datasets and examine the performance of our classifier under different settings. We vary the following parameters:.

1. **The number of documents in the corpus** (N): We consider two setups, 1000 and 10000 documents.

2. **The size of vocabulary** (V): We consider three setups, 500, 2500, and 5000 words.

3. **The number of words per document**: (n_i): We generate n_i for $i = 1, \ldots, N$ from Poisson distribution with three different means (10, 50, and 100). This means that the average number of words per document is 10, 50, and 100, respectively.

4. **Proportion of documents in the positive class** (π): We consider three setups, 5%, 10%, and 50%.

5. **Difficulty of classification** (β): To control the difficulty of classification, we change the way the vector of observing words for each class ($\eta_{\cdot 0}$ and $\eta_{\cdot 1}$ for negative and positive class, respectively) are generated from the original model. Note that simulating real corpora poses a challenge due to the presence of words that occur in both positive and negative classes, as well as words that are more prevalent in one class compared to the other. For instance, words like "election" and "democracy" are expected to appear frequently in documents labeled as "political" compared to those labeled as "non-political."

   To mimic this scenario, we first draw a "base" parameter vector, denoted as $\tilde{\eta}$, from a Dirichlet distribution with parameter β. We set $\eta_{\cdot 0} = \eta_{\cdot 1} = \tilde{\eta}$. Subsequently, we exchange the 10 highest values with those associated with the 10 lowest values in the

vector $\eta_{\cdot 1}$ while leaving the vector $\eta_{\cdot 0}$ unchanged. This results in $\eta_{\cdot 1}$ and $\eta_{\cdot 0}$ being nearly identical, except for the swapped elements.

Consequently, the 10 swapped words act as keywords since their values in $\eta_{\cdot 1}$ are entirely distinct from their values in $\eta_{\cdot 0}$. This allows us to control the level of informativeness of the keywords when manipulating the values in β. For instance, lower values in β concentrate the values in η towards the edges of the probability space, thereby making the keywords more informative. Conversely, higher values in β distribute the values in η more evenly, resulting in less informative keywords and a more challenging classification task.

We investigate three different values for the hyperparameter β in the Dirichlet distribution: 0.1, 0.5, and 0.9. As β is a vector with a length of V (the vocabulary size), for each simulation setting we fix all elements of β to the same value. Figure M.1 illustrates how the values in the vector η are generated under the different values of β. As the figure reveals, smaller (larger) values for β result in more (less) informative keywords, and, as a result, easier (more difficult) classification tasks.

In our study, we examine a total of 162 distinct simulation setups, obtained by combining different values for five parameters: two choices for the first parameter (N), three choices for the second (V), third (n_i), fourth (π), and fifth (β) parameters. For each setup, we generate 100 datasets and calculate the average classification metrics. In this discussion, we primarily focus on the F1 score as the primary metric. However, we also provide the results for accuracy, precision, and recall in a supplementary appendix dedicated to our simulation analysis. It is important to note that all metrics are evaluated on out-of-sample data, ensuring that the classifier is assessed on data it has not been trained on.

M.2 Results

Figure M.2 presents a summary of our simulation results through QQ-plots, which illustrate the out-of-sample F1 scores obtained from random sampling and active learning. We calculate the average out-of-sample F1 score for each simulation setup using both sampling methods and compare their performance. The QQ-plots in each panel depict the F1 scores at different proportions of hand-labeled documents in the corpus: 0.1, 0.5, and 0.9, from left to right. The colors in the plots represent various proportions of the positive class in the corpus: 0.05 (red), 0.1 (orange), and 0.5 (black). Across all simulation setups, we consistently observe that active learning outperforms random sampling, as indicated by the QQ-lines positioned above the diagonal line.

Our findings reveal that active learning performs well when dealing with imbalanced data and a limited number of hand-labeled documents. Comparing the different panels, we notice

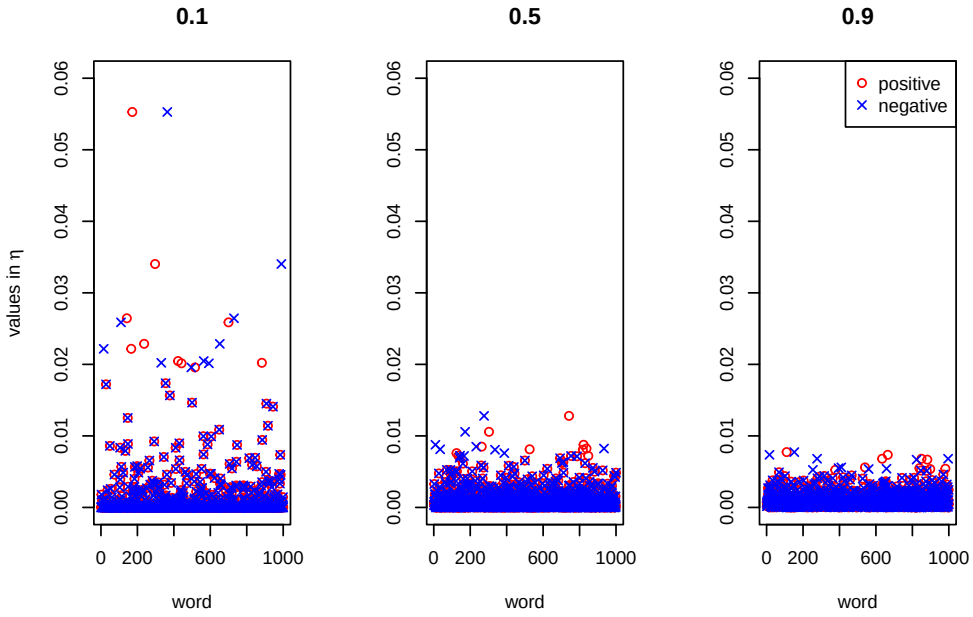

Figure M.1: η **values for three different simulation setups.**
The figure displays the word indices on the horizontal axis and the corresponding η values for each word on the vertical axis. Three values of β are used: 0.1 (left), 0.5 (middle), and 0.9 (right). In the graph, circles represent values associated with the positive class, while crosses represent values associated with the negative class. The vocabulary consists of 1000 unique words, and we swap the 10 highest values in $\eta_{\cdot 1}$ with the 10 lowest values in $\eta_{\cdot 1}$. From the figure, it is evident that smaller (larger) values of β lead to more (less) informative keywords. Consequently, the classification tasks become easier (more difficult).

that the QQ-lines in the left panel significantly deviate from the diagonal line, indicating the superior performance of active learning when only a small proportion of documents are labeled by hand. As we increase the number of hand-labeled documents, the benefit derived from active learning diminishes, as demonstrated by the QQ-lines in the right panel aligning more closely with the diagonal line.

Furthermore, when comparing the different QQ-lines, we observe that the lines representing imbalanced classification classes (red and orange) are situated above the diagonal line, while the line representing the balanced corpus (black) closely aligns with the diagonal line. This observation suggests that active learning is particularly advantageous when dealing with imbalanced classes.

Finally, we present results about the (in-sample) statistical properties of the key parameters and (out-of-sample) predictive performance of our proposed method. We specifically focus on two simulation settings: an easy setting where 20 keywords associated with a class are present, and a hard setting where there are only 5 keywords are present. The number

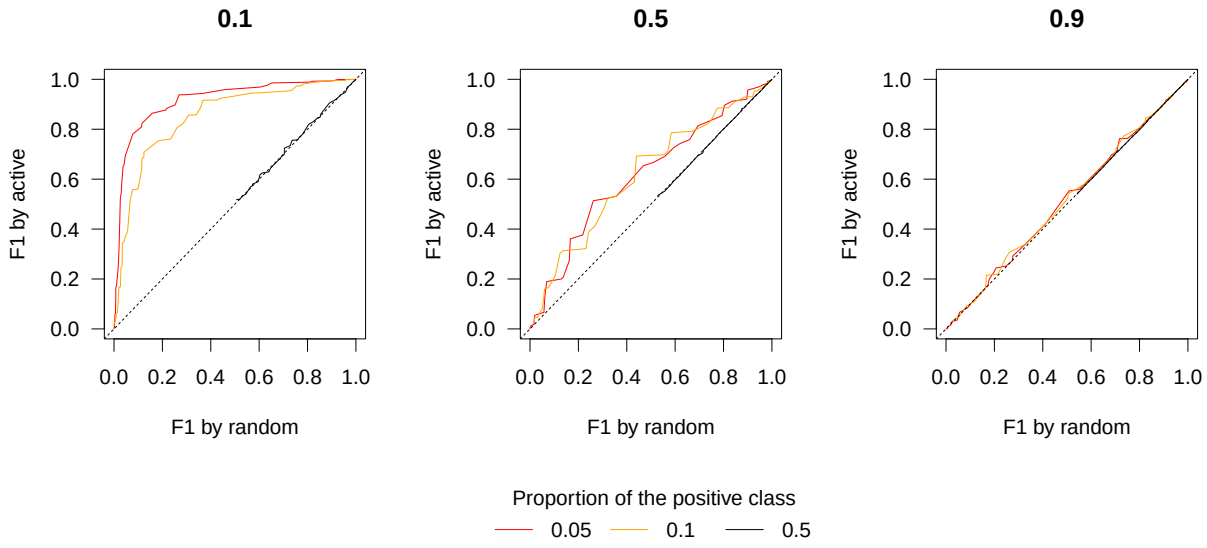

Figure M.2: **Out-of-sample F1 scores between random sampling and active learning**
This figure illustrates the QQ-plot for out-of-sample F1 scores in different simulation setups. Each panel represents a different proportion of hand-labeled documents in the corpus, ranging from 0.1, 0.5, to 0.9 from left to right. The colors distinguish various proportions of the positive class, namely 0.05 (red), 0.1 (orange), and 0.5 (black). Across all simulation setups, active learning consistently outperforms random sampling as indicated by the QQ-lines lying above the diagonal line. The figure shows that across simulation settings, active learning performs well in scenarios with imbalanced data (orange and red lines) and when only a small portion of the corpus can be hand-labeled (left panel).

of keywords also controls the difficulty of classification, as more keywords provides more information. The results for the easy and hard cases are presented in Figure M.3 and Figure M.4, respectively. Each figure is divided into three panels. The leftmost panel presents the Mean Square Error (MSE) of π (the overall probability of finding a positive document), the middle panel shows the MSE of η (the vector of observing each word in a given class), and the rightmost panel the out-of-sample F1 score. Our results show that as the number of hand-labeled documents increases, both in active and passive sampling schemes, the MSE of η and π decreases. Although active learning at earlier stages results in a slightly larger MSE for π due to the possible bias induced by labeling the most uncertain cases first,[13] it is this process of effective annotation that ultimately appears to improve predictive performance. These findings confirm our previous conclusions (via validation and reanalysis of

[13]See Farquhar et al. (2021) for an thorough discussion of the tradeoffs between statistical bias induced by active learning and predictive performance, especially in supervised learning.

published work) that active learning, particularly in the early stages of the labeling process, outperforms passive approaches.

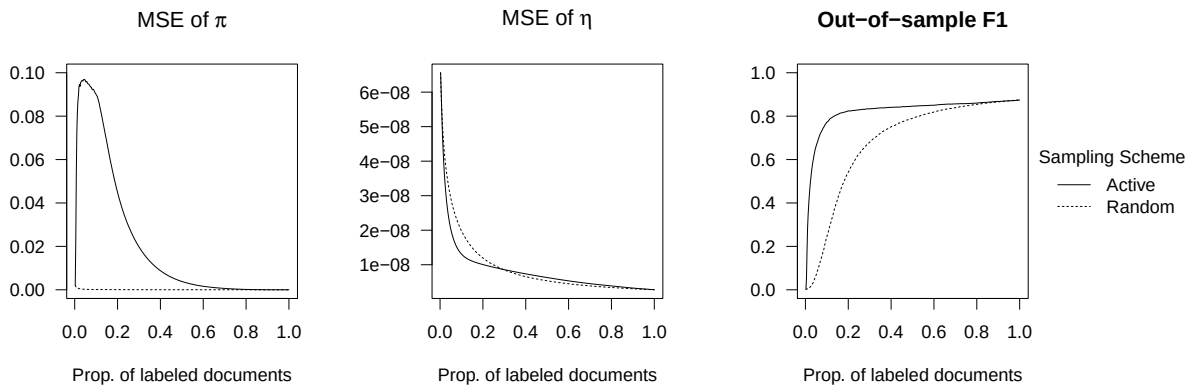

Figure M.3: **Mean Square Error (MSE) of π and η between active and passive sampling (Easy classification)**
These figures showcase how the bias in the parameter and the classification performance changes across iterations between active and passive sampling with a simulation dataset on which the classification is easy for both sampling scheme. The simulation dataset has 10000 documents, 5000 unique words, 50 words per document on average, 10% of the documents are positive, and $\beta = 0.5$. To make the classification easy, we swapped 20 words with the highest values in $\eta_{\cdot 1}$ with the 20 words with the lowest values in $\eta_{\cdot 1}$. The left and the middle figures illustrate the MSE in π and η across labeling iterations. The right figure shows the out-of-sample F1 score. The solid line is for the active sampling and the dashed line is for the passive sampling. Note that the MSE for π is small but non-zero for the random sampling. With active learning, the MSE in π increases at the early part of the labeling iterations. However, the out-of-sample F1 score keeps increasing when the bias in π is large. The bias in π and η decreases as the number of hand-labeled documents increases.

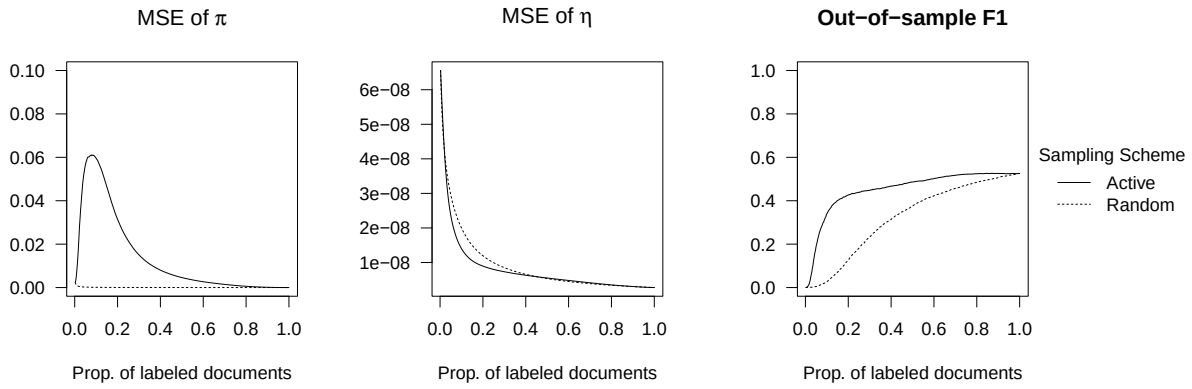

Figure M.4: **Mean Square Error (MSE) of π and η between active and passive sampling (Hard classification)**
Similarly to Figure M.3, this figure shows the results with a simulation dataset on which the classification is difficult for both sampling scheme. To make the classification difficult, we swapped only 5 words with the highest values in $\eta_{\cdot 1}$ with the 5 words with the lowest values in $\eta_{\cdot 1}$. Again, the MSE of π with active learning increases at an early stage, but the MSE of both π and η decreases as the number of hand-labeled documents increases.

References

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

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

Dempster, A., Laird, N., and Rubin, D. (1977), "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1), 1–22.

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.

Farquhar, S., Gal, Y., and Rainforth, T. (2021), On Statistical Bias In Active Learning: How and When to Fix It,, in *International Conference on Learning Representations*.
URL: *https://openreview.net/forum?id=JiYq3eqTKY*

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

Greene, D., and Cunningham, P. (2006), Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering,, in *Proc. 23rd International Conference on Machine learning (ICML'06)*, ACM Press, pp. 377–384.

Larsen, M. D., and Rubin, D. B. (2001), "Iterative Automated Record Linkage Using Mixture Models.," *Journal of the American Statistical Association.*, 96(453), 32–41.

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.

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.

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.

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.