

# Appendix: Automated Text Classification of News Articles: A Practical Guide

**Pablo Barberá**   **Amber E. Boydston**   **Suzanna Linn**  
USC                      UC Davis                      Penn State

**Jonathan Nagler**   **Ryan McMahon**  
NYU                      Google

## Contents

<b>1</b>	<b>Coding Instructions</b>	<b>1</b>
<b>2</b>	<b>Details for the Training Data Used in the Analysis</b>	<b>5</b>
<b>3</b>	<b>Comparing the Subject-Category and Keyword Corpora</b>	<b>6</b>
<b>4</b>	<b>Selecting a Classifier</b>	<b>9</b>
<b>5</b>	<b>Selecting the Unit of Analysis</b>	<b>13</b>
<b>6</b>	<b>Choosing Between Types of Coders: Low Quality and Low Cost vs. High Quality and High Cost</b>	<b>18</b>
<b>7</b>	<b>Allocating Total Codings: More Documents vs More Coders</b>	<b>23</b>
<b>8</b>	<b>Dictionaries vs SML: Accuracy and Precision in UG Truth</b>	<b>26</b>
<b>9</b>	<b>Convergent Validity</b>	<b>27</b>
<b>10</b>	<b>SML Classifier Performance as a Function of the Size of the Training Dataset</b>	<b>30</b>

# 1 Coding Instructions

The 5-category instructions for undergraduates and CrowdFlower participants for coding sentiment are detailed below, followed by the 9-point instructions given to CrowdFlower coders.

## 5-Category Coding

In this job, you will be presented with excerpts from newspaper articles. We will ask you to read each excerpt and then answer a few questions about whether the excerpt tells you anything about how the United States economy is doing.

Process

1. Read the article
2. Determine if the article gives you an indication of how the economy is performing.

For example, an article with this headline probably gives you an indication of how the economy is performing: "Unemployment just went up by 2%."

For each article, your job is to judge whether it gives YOU an indication about how the economy is performing.

3. If it gives you an indication of how the economy is performing, tell us whether that indication is positive, neutral, negative, or mixed – or you are not sure.

If the text contains typos or is otherwise corrupted in some way, do your best to make out the message of the article.

Thank You!

Thank you very much for your work!

**[NEW PAGE ON CROWDFLOWER]**

Read the highlighted segment below paying close attention to detail:

Headline: Administration Using Study To Push Elderly Drug Plan: Pressing Republicans to Reach Compromise

Date: 2000-04-26

WASHINGTON, April 25 – Prices for the 50 drugs most frequently used by older Americans increased last year at nearly twice the rate of inflation, a study to be issued on Wednesday at the White House says. The study found a similar trend over the last six years. President Clinton planned to cite the data in arguing that drug costs constitute a growing burden for the elderly and that Congress should pass legislation to reduce the burden. The Democratic leaders of Congress, Senator Tom Daschle of South Dakota and Representative Richard A Gephardt of Missouri, intend to join Mr Clinton at the White House. The three will press Republicans in Congress to reach a compromise and make drug coverage available to all 39 million Medicare beneficiaries.

Q1) Does the article provide some indication about how the U.S. economy is performing?

- Yes
- No
- Not Sure

Q2) If so - is the indication:

- Positive
- Negative
- Mixed
- Neutral
- Not Sure

## 9-Point Coding

In this job, you will be presented with excerpts from newspaper articles. We will ask you to read each excerpt and then answer a few questions about whether the excerpt provides information about how the US economy is doing.

If you complete a three-question survey and code more than 10 articles, we will award you an extra bonus of \$0.50.

### Process

1. Read the article
2. Determine if the article gives an indication of how the United States economy is performing.

For example, an article with this headline probably gives you an indication of how the U.S. economy is performing: "Unemployment just went up by 2%."

Note that you are only being asked whether the article tells you something about how the United States economy is doing – so articles that only contain information about the economy in other countries are not relevant.

3. If the article gives you an indication of how the economy is performing, tell us whether the indication is negative or positive, using a scale ranging from 1 (very negative) to 9 (very positive). You can pick a value anywhere on the scale to indicate where the article is on the scale.

If the text contains typos or is otherwise corrupted in some way, do your best to make out the message of the article.

### Bonus

To receive an extra \$0.50 bonus, please complete the survey in this link (will open in a new window). The survey contains three simple questions and should take only 30 seconds to complete. If you complete this survey and code at least 10 articles, we will award you the bonus within the next 72 hours once this task is completed.

Thank You!

Thank you very much for your work!

**[NEW PAGE ON CROWDFLOWER]**

Read the highlighted segment below paying close attention to detail:

Headline: Nation's Governors See a Dismal Economic Outlook and a Slow Recovery

Date: 2009-02-22

WASHINGTON – *The nation's governors said Saturday that passage of a \$787 billion bill to stimulate the economy might help them avert draconian budget cuts, but that they did not expect to see signs of an economic recovery until late this year or early 2010. The officials, arriving here for the winter meeting of the National Governors Association, said that state revenues were coming in far below their projections and that the new federal measure, while helpful, would not be a panacea. Gov. Jon Huntsman Jr. of Utah, where the economy is better than in most states, said the revenue figures were "still dismal". Asked when the recovery would start, Mr. Huntsman, a Republican, said: "We were hoping in the fourth quarter of this year." Gov. Steven L. Beshear of Kentucky, a Democrat, said: "If the experts are correct, next year may be even worse than this year. I think very probably they are correct."*

Q1) Does the article provide some indication about how the U.S. economy is performing?

- Yes
- No
- Not Sure

Q2) If so - is the indication:

- |                       |                       |                       |                       |                       |                       |                       |                       |                       |
|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| <input type="radio"/> |
| 1                     | 2                     | 3                     | 4                     | 5                     | 6                     | 7                     | 8                     | 9                     |
| Negative              |                       |                       |                       |                       |                       |                       | Positive              |                       |

## 2 Details for the Training Data Used in the Analysis

Table 1: Details for Training Data Used in the Analysis

Data ID	Analysis (Section)	Unique Objects <sup>a</sup> (Unit of Analysis)	Number & Type of Coders <sup>b</sup>	Number of Objects in Training Data	Avg. Number of Coders per Object <sup>c</sup>	Coding Scale <sup>d</sup>	Truth Dataset <sup>e</sup>
1AC <sup>f</sup>	Unit of Analysis (Section 3.1)	10,000 (S)	3 CF	8,642	2.16	9-point scale	CF Truth
1SC		2,000 (A)	3 CF	1,790	2.26	9-point scale	CF Truth
5AC	Dictionary v SML (Section 4)	4,400 (A)	3-10 CF	4,070	2.51	9-point scale	UG Truth CF Truth
UG Truth		4,195 (S)	2-14 UG	250 (A)	2.13	5-category	NA
CF Truth		4,400 (A)	10 CF	442 (A)	7	9-point scale	NA
2SU <sup>g</sup>	Coder Quality (Appendix Section 6)	(S)	1-8 UG	420	2.91	5-category	NA
2SC <sup>h</sup>		(S)	3-6 CF	420	2.46	5-category	NA
3SU		4,195 (S)	2-14 UG	1,945	2.13	5-category	NA
4SC		3,885 (S)	3-6 CF	3,136	1.77	5-category	NA

Note: All articles in the training data were randomly selected over the period 1947-2014. Data ID names are such that the number denotes the corpus, the first letter denotes the unit of analysis, and the second letter identifies the coder pool. The machine learning algorithm used to train the classifier uses logistic regression with an L2 penalty, where the features are the 75,000 most frequent stemmed unigrams, bigrams, and trigrams appearing in at least 3 documents and no more than 80% of all documents (stopwords are included). The two 4,400 article datasets comprise distinct datasets.

<sup>a</sup> Sample size denotes the number of objects available to be coded. (S) denotes sentences (using the first five sentences of an article, individually presented) and (A) article-segments, where an article-segment is defined as the first 5 sentences of an article presented in order.

<sup>b</sup> All CrowdFlower (CF) coders were located in the U.S. (UG) coders were Penn State undergraduate students trained as a group.

<sup>c</sup> The average number of coders per object who coded the tone of the object (excluding objects coded as irrelevant).

<sup>d</sup> The 9-point scale ranged from 1 (very negative) to 9 (very positive). Categories were collapsed such that 1-4=0 (negative), 6-9=1 (positive). We dropped responses at the midpoint (5) since earlier tests with a multinomial classifier suggested keeping them would lead to noisier estimates. The 5-category coding scheme allowed coders to label a sentence as negative, mixed, neutral, not sure, positive. Responses were recoded to -1 (negative), 0 (mixed, neutral, not sure), and +1 (positive) before computing the variance.

<sup>e</sup> UG Truth is based on Dataset 2SU. The "truth" in UG Truth is based on students with high agreement. The "truth" in CF Truth is based on 70% or higher agreement among coders.

<sup>f</sup> All the 10,000 sentences in dataset 1SC come from the same 2,000 articles in dataset 1AC.

<sup>g</sup> Dataset 2SU is the overlap of 3SU and 4SC.

<sup>h</sup> Dataset 2SC is the overlap of 4SC and 3SU.

### 3 Comparing the Subject-Category and Keyword Corpora

Data for Figure 1 in the text is presented in detail in Table 2 below. In general, in years where the keyword corpus contained relatively more articles, so, too, did the subject category corpus ( $\rho = .71$ ). But in some years the keyword corpus contained over three times as many articles as the subject category corpus, while in others both corpora contained similar counts, and finally, in two years (2002 and 2011), the subject category corpus contained slightly more articles. Notably, there is a downward trend in the number of articles in the keyword corpus that is not apparent in the subject category corpus.<sup>1</sup>

---

<sup>1</sup>Due to discrepancies in the ‘cleanliness’ and formatting of the texts, along with differences in meta-data accuracy (e.g., date published) an exact matching approach to identifying overlap in the two corpora would dramatically bias estimates of congruency downward. Instead, we generated a set of potential matches by searching for articles with similar headlines and published in the same year. (Article headline similarity was defined as having a maximum Levenshtein edit distance of 0.35.) This produced a set of articles from the subject category corpus that were potential matches for each article in the keyword corpus (we constrained the sets to have a maximum of ten potential matches). We then randomly sampled 25% of all potential matches (2,600 pairs). These sampled article pairs were subsequently coded as being true or false matches and then used as training data for an AdaBoost classification tree algorithm using the caret package in R. We were able to achieve a 10-fold cross-validated accuracy of  $\sim 99.3\%$ . Baseline accuracy over the entire training set was 58.7%. The resulting model was used to classify headlines as unique or matched for the remaining 75% of potential matches. Unique articles could be labeled as matches in multiple sets of article pairs. Pairs of articles containing non-unique article IDs were cleaned by hand. This form of duplication was exceedingly rare; only 145 of 4,368 pairs of articles contained a non-unique article identifier.

Table 2: Comparing the Subject Category Corpus with the Keyword Corpus: Total, Unique, and Overlapping (Common Corpus) Article Counts from the *New York Times*, 1980-2011

Year	Keyword Corpus	Subject Category Corpus	Common Corpus	Unique Keyword	Unique Subject Category
1980	1767	516	73	1694	443
1981	1545	945	133	1412	812
1982	1960	1361	344	1616	1017
1983	1618	840	206	1412	634
1984	1304	629	112	1192	517
1985	1103	481	85	1018	396
1986	1020	444	84	936	360
1987	1340	552	93	1247	459
1988	1125	521	105	1020	416
1989	1016	522	139	877	383
1990	862	587	98	764	489
1991	1452	972	263	1189	709
1992	1243	925	211	1032	714
1993	962	607	125	837	482
1994	1076	588	138	938	450
1995	736	484	91	645	393
1996	769	415	65	704	350
1997	840	437	87	753	350
1998	742	471	107	635	364
1999	804	436	102	702	334
2000	608	451	83	525	368
2001	763	865	140	623	725
2002	556	558	111	445	447
2003	502	463	96	406	367
2004	545	369	99	446	270
2005	474	277	90	384	187
2006	501	325	105	396	220
2007	505	282	77	428	205
2008	784	645	179	605	466
2009	988	883	312	676	571
2010	815	564	221	594	343
2011	462	480	116	346	364
Total	30787	18895	4290	26497	14605

Note: Cell entries are annual counts of articles retrieved for each corpus. See text for details explaining the generation of each corpus. See Footnote 1 for a description of the methods used to calculate article overlap.

We report the relevance results based on the full set of worker-level codings (not aggregated to the article level by taking the model response) in Table 3. These are nearly identical to the results after aggregating all the coder responses to the article level reported in Section 2 of the paper.

Table 3: Proportion of Relevant Articles by Corpus

Relevance	Articles in both		
	SSW and Ours	Unique Ours	Unique SSW
Not Relevant	0.55	0.58	0.60
Not Sure	0.01	0.01	0.01
Relevant	0.45	0.42	0.40

Note: Cell entries indicate the proportion of sentences in each dataset (and their overlap) coded as providing information about how the US economy is doing. One thousand articles from each dataset were coded at the sentence level by three CrowdFlower workers located in the US. Each coder was assigned a weight based on her overall performance before computing the proportion of sentences deemed relevant. If two out of three (weighted) coders concluded a sentence was relevant, the aggregate response is coded as “relevant”.

## 4 Selecting a Classifier

### Preprocessing Decisions

Typically prior to training a classifier the analyst makes a number of text processing decisions including whether to stem the text, how to handle stop words, and the nature and number of features of the text to include. We consider these in turn.

First the analyst must decide whether to stem the text (truncate words to their base). When we truncate words we are asserting that words with the same base mean the same thing (i.e., ‘cars’ is the same as ‘car’), and thus they constitute a single feature. The main benefits of stemming relate to the reduction of two things: *i*) dimensionality and *ii*) loss of relevant terms. Without stemming these words, ‘car’ and ‘cars’ would appear as separate features, which can increase the size of the document-term matrix immensely. This is especially true when using anything greater than unigrams.<sup>2</sup>

Further, by combining words of the same root we can generally reduce the likelihood of throwing out potentially relevant words. When we stem words, we are only changing the overall sparsity of the matrix by a small amount, while simultaneously decreasing the relative number of potential parameters to fit by a much larger amount.<sup>3</sup> While the increase in potential parameters can be easily solved by limiting the feature-set to the top  $K$  words, this can be problematic for the exclusion of relevant terms. If ‘car’ is the 28,000<sup>th</sup> most common word and ‘cars’ is the 42,000<sup>th</sup> most common word, neither will be included in a feature-set of the top 25,000 terms, even though the concept of ‘car’ may be a relevant one.

This is not to say that one should always use stemming as a pre-processing technique. In certain cases stemming can actually do more harm than good. Take, for in-

---

<sup>2</sup>Using only unigrams in our NYT data, stemming reduces the number of features by roughly 60,000 (~ 4.5%). This results in 3.7B fewer cells in the document-term matrix.

<sup>3</sup>For the same data as in the previous footnote, stemming increases the sparsity of the document-term matrix by just ~ 0.0002%, but reduces the number of features by ~ 4.5%.

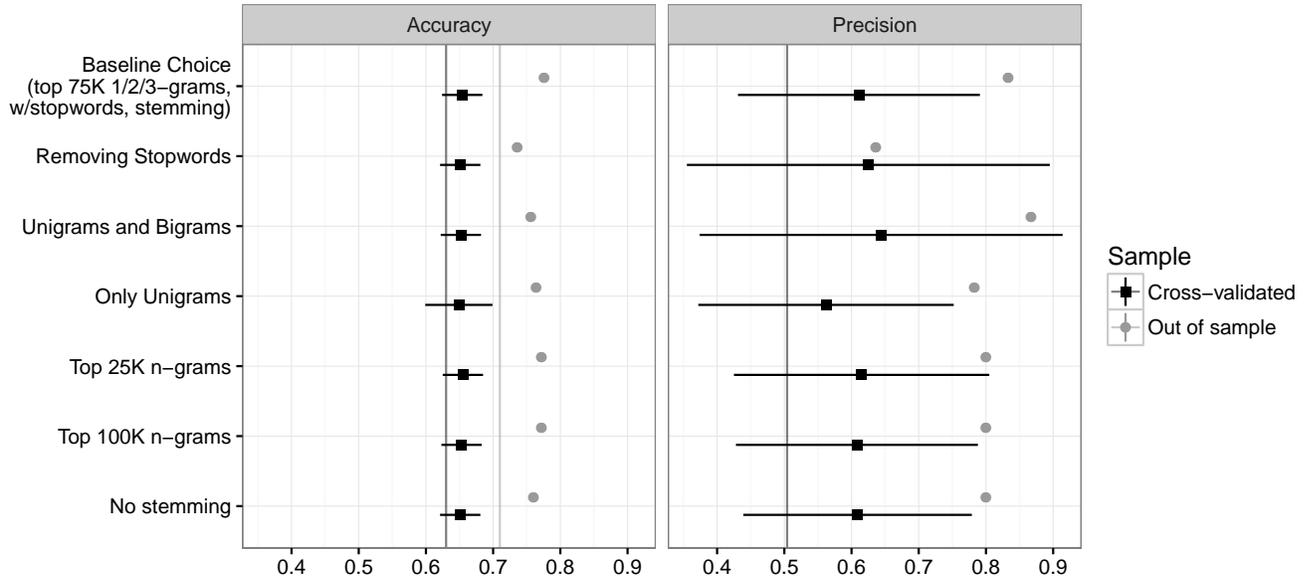
stance, a hypothetical question about prospective *vs.* retrospective media coverage following presidential addresses. Here the tenses of verbs would be key to the analysis, since they are certainly features that would have distinctly different distributions across the two classes (past and future). By stemming the verbs in such an analysis we would be throwing away very valuable information.

Second, the analyst needs to decide how to deal with stop words (commonly used words that do not have contain relevant information). Should all stop words from a general list be omitted from the training data, or should the stop words be tailored to the specific task? For instance, in our task of classifying sentiment on the economy, the words 'down' and 'up' have obvious meaning. But these words are considered stopwords to be dropped before analysis by commonly used text processing software such as scikit learn. Throwing out 'down' from a sentence with "unemployment has gone down" is obviously not going to improve our classifier.

Third, the analyst needs to decide the nature and number of features to include. For example, the analyst has to decide whether to include just single words (unigrams), or to include bigrams or trigrams, or other combinations. If one is coding for topic, it might not be as important to include bigrams and trigrams. However, in coding for economic sentiment, where "rising unemployment" is completely different than "falling unemployment" - inclusion of bigrams seems essential. We can also go beyond N-grams to think about co-occurrence within sentences. "Falling" does not need to immediately proceed "unemployment" to suggest that the sentence talks about "falling unemployment", if it is in the same sentence, we can infer that it is likely to be modifying unemployment.

Based on these considerations we adopted a baseline set of decisions in which we used 75,000 n-grams appearing in at least 3 documents and no more than 80% of all documents, using both stopwords and stemming. We compared results from this baseline with six alternative preprocessing decisions: a) removing stopwords, b) using only unigrams

Figure 1: Performance of Classifiers Depending on Feature Selection



Note: Vertical lines indicate baseline accuracy and precision (predicting the modal category). Performance metrics are reported using cross-validation (in-sample, 10 folds) and on the ground truth dataset (out-of-sample).

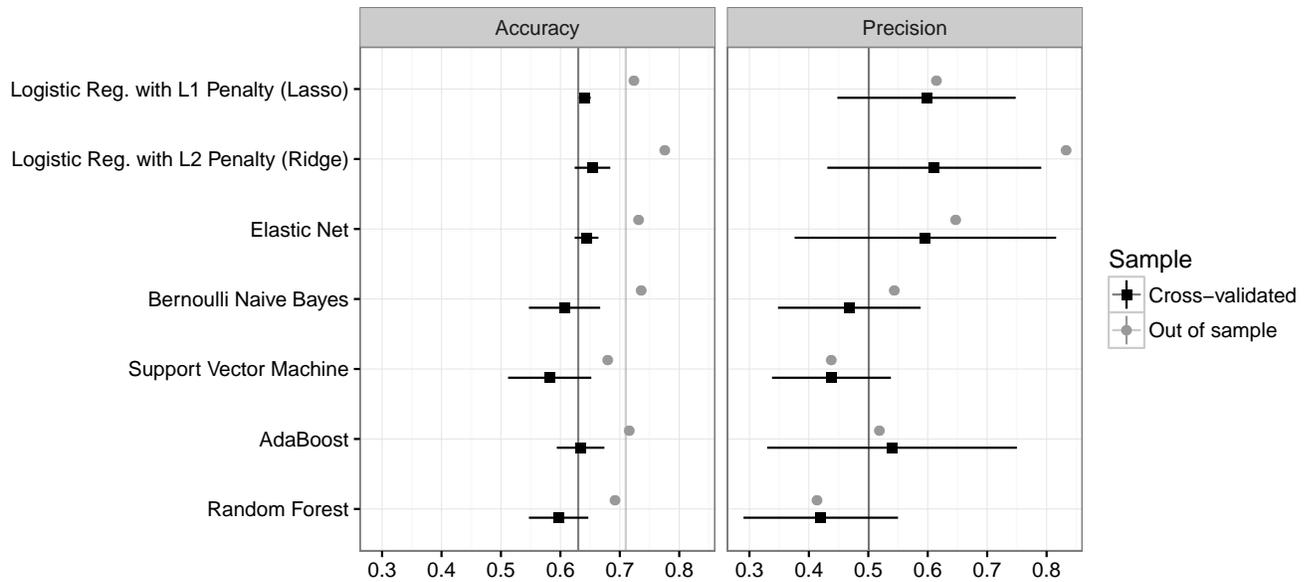
and bigrams, c) using only unigrams, d) using only the top 25,000 n-grams, e) using up to 100,000 top n-grams, and f) not stemming the text of the articles. We assess classifier performance using accuracy and precision measured both using 10-fold cross-validation, and in our ground truth dataset coded by crowd workers with high agreement. Measures of accuracy and precision using both techniques are reported in Figure 1. We do not find any large differences in accuracy or precision as a function of feature selection. Our baseline approach appears to combine high values on all the different performance metrics.

### Comparing Classifiers

In addition, we report the results for accuracy and precision of a number of other common machine learning classifiers using our baseline decisions for feature selection in Figure 2.

A logistic regression classifier with L2 penalty (Ridge Regression) appears to yield the best performance and is significantly faster to estimate than ensemble classifiers, such as AdaBoost and Random Forests.

Figure 2: Performance of Different Classifiers



Note: Vertical lines indicate baseline accuracy and precision (predicting the modal category). Performance metrics are reported using cross-validation (in-sample, 10 folds) and on the ground truth dataset (out-of-sample).

We note that all of these classifiers are one-stage classifiers: we do not attempt to combine the classification of relevance and tone here; but rather code all articles for tone. This is treating irrelevant articles as having tone. We experimented with developing a classifier for relevance (all articles in the training dataset were first coded for relevance, then only coded for tone if the coder stated that the article was relevant), and applying this classifier to filter out articles that were not relevant. What we were most worried about with any such procedure is generating selection bias: if our relevance filter eliminated more relevant negative articles or positive articles (or vice-versa), we would be introducing bias, when the intent was to remove noise. We found that we were not introducing bias - but have not yet further tested the efficacy of the filter.

## 5 Selecting the Unit of Analysis

As discussed in section 3.1 of the paper, whether coding based on sentences or articles—or in our case, article-segments—is best likely depends on at least four features of the data: 1) the size of the vocabulary, 2) the distribution of positive and negative features within sentences, 3) the distribution of positive and negative sentences within articles/article-segments and 4) the proportion of sentences that are not relevant to tone that reside in articles/article-segments coded as positive or negative. In Table 4, we first present the distribution of positive and negative sentences within article-segments coded as negative and positive based on unanimous agreement of three coders as reported in Section 3.1. Then in Tables 5-8, we present this distribution allowing for disagreement among coders. We report the same information allowing for disagreement among coders in Table 9.

Table 4: Article-Segment by Sentence Tone

	<b>Average # of Positive Sentences</b>	<b>Average # of Negative Sentences</b>
<i>Positive Article-Segments</i>	0.91	0.27
<i>Negative Article-Segments</i>	0.08	1.00

*Note:* Analysis is based on Datasets 1SC and 1AC, Appendix Table 1. Article-segments are counted as positive or negative here only if there was unanimity of the article-segment-level coding by the 3 coders. This coding rule results in 85 positive article-segments and 225 negative article-segments. The average number of positive (negative) sentences represents the number of sentences that all 3 coders coded positive (negative) for each article-segment.

Table 5: Article-Segment by Sentence Tone (2/3) Cut-Off

	<b>Average # of Positive Sentences</b>	<b>Average # of Neutral or Irrelevant Sentences</b>	<b>Average # of Negative Sentences</b>
<i>Positive Article-Segments</i>	1.60	2.43	0.95
<i>Negative Article-Segments</i>	0.54	2.47	1.96

*Note:* This coding rule results in 317 positive article-segments and 667 negative article-segments. The average number of positive (negative) sentences represents the number of sentences that at least 2 of 3 coders coded positive (negative) for each article-segment coded as either positive or negative by at least 2 of 3 coders. Remaining sentences are coded as neutral or irrelevant.

Table 6: Article-Segment by Sentence Tone; Averaging & (2/3) Cut-Off

	<b>Average # of Positive Sentences</b>	<b>Average # of Neutral or Irrelevant Sentences</b>	<b>Average # of Negative Sentences</b>
<i>Positive Article-Segments</i>	1.79	1.97	1.22
<i>Negative Article-Segments</i>	0.83	2.07	2.07

<sup>1</sup> *Note:* This table is similar to Table 5, but the values here are the average of a within article-segment average across coders. For example, imagine there are two positive article-segments, call them  $A_1$  and  $A_2$ ; further, assume that for each article-segment there are 5 sentences and 3 coders. Finally, say that we are interested in the top left cell (positive sentences in positive article-segments). For  $A_1$ , we will take the number of sentence codings greater than 5, and divide that by the number of coders (3). We do the same for  $A_2$ . Then to produce the value appearing in the table, we average those results from  $A_1, A_2$ .

<sup>2</sup> This coding rule results in 317 positive article-segments and 667 negative article-segments.

<sup>3</sup> The (2/3) cut-off rule was only applied to article-segments.

Table 7: Article-Segment by Sentence Tone; Averaging & (3/3) Cut-Off

	<b>Average # of Positive Sentences</b>	<b>Average # of Neutral or Irrelevant Sentences</b>	<b>Average # of Negative Sentences</b>
<i>Positive Article-Segments</i>	2.22	1.58	1.16
<i>Negative Article-Segments</i>	0.77	1.80	2.40

<sup>1</sup> *Note:* See Table 6 (Note 1) for a description of the process.

<sup>2</sup> This coding rule results in 85 positive article-segments and 225 negative article-segments.

<sup>3</sup> The 3/3 cut-off rule was only applied to article-segments.

The number of irrelevant sentences in article-segments that convey tone is also likely to affect the performance of classifiers based on sentence-level coding compared to classifiers based on article-segment-level coding. If article-segments contain a large number of irrelevant sentences in an otherwise toned text, an article-segment classifier would treat those features as informative for article classification while a sentence classifier would disregard them as uninformative. For example, imagine there’s an article-segment about Detroit, and only the fifth sentence mentions “and unemployment is at an all-time high”. The sentence-level classifier would only learn from this sentence, but the article-segment-level classifier might identify other features that appear in the first four sentences that also predict negative economic tone (e.g., ‘Detroit’, ‘Michigan’) even if a coder would not identify them as such. We see this problem in our article-segment-level coding when dates show up as predictive features. Note, however, this only happens if these irrelevant features are correlated with relevant features. Further, the higher the proportion of irrelevant sentences, the more noise is added to the article-segment feature matrix. In Table 8 we report the average number of sentences coded as relevant and irrelevant given that the article-segment was unanimously coded as relevant or irrelevant by all three coders. We find that on average slightly more sentences are coded as irrelevant by all 3 coders (2.64) as opposed to relevant (2.33) in article-segments coded as relevant, while article-segments coded as irrelevant averaged 3.64 irrelevant sentences and 1.35 relevant sentences.

Table 8: Article-Segment by Sentence Relevance (3/3) Cut-Off

	Average # of Relevant Sentences	Average # of Irrelevant Sentences
<i>Relevant Article-Segments</i>	2.33	2.64
<i>Irrelevant Article-Segments</i>	1.35	3.64

*Note:* This coding rule results in 812 relevant article-segments and 1,151 irrelevant article-segments. Cell entries represent the average number of sentences all 3 coders identified as relevant or irrelevant in article-segments coded as relevant or irrelevant.

Table 9: Article-Segment by Sentence Relevance (2/3) Cut-Off

	<b>Average # of Relevant Sentences</b>	<b>Average # of Irrelevant Sentences</b>
<i>Relevant Article-Segments</i>	3.68	1.30
<i>Irrelevant Article-Segments</i>	2.22	2.76

*Note:* This coding rule results in 1,456 relevant article-segments and 507 irrelevant article-segments. Cell entries represent the average number of sentences at least 2 of 3 coders identified as relevant or irrelevant in article-segments coded by at least 2 of 3 coders as relevant or irrelevant.

As a final diagnostic, we calculated the density of the sentence and article-segment feature matrixes used to classify the text. One possibility is that removing the large number of sentences coded as irrelevant or neutral will reduce the relative sparsity (increase the density) of the sentence feature matrix enough to lessen this threat to efficiency relative to the article-segment feature matrix. Table 10 shows the density (inverse of sparsity) of the article-segment and sentence feature matrixes in the full training dataset (including irrelevant objects) and in the data used to train the classifier, after having removed the irrelevant objects. As can be seen, dropping irrelevant objects reduces the sparsity of both feature matrixes – and does so more for the sentence feature matrix – but relative to the sentence feature matrix, the article-segment feature matrix is still nearly 3 times more dense. Similarly, the variance of the sentence feature matrix used to train the classifier is considerably smaller (0.0037) than that of the article-segment feature matrix (0.0221). Thus while removing irrelevant objects reduces sparsity, even with a relatively large number of irrelevant sentences, any gains in specificity of coding at the sentence level may still be swamped by the lack of variance in features across sentences.<sup>4</sup>

Table 10: Feature Set Size and Density (CF)

		Density <sup>1</sup>	Features	Observations <sup>2</sup>	Variance <sup>3</sup>
<b>Article-Segment</b>	<i>Full</i> <sup>4</sup>	0.69%	33,640	1,963	0.0139
	<i>Classifier</i>	1.07%	20,563	1,181	0.0221
	% $\Delta$	55.69%	–38.87%		
<b>Sentence</b>	<i>Full</i>	0.16%	34,236	9,774	0.0021
	<i>Classifier</i>	0.29%	18,503	5,062	0.0037
	% $\Delta$	78.95%	–45.95%		

*Note:* Values of the cell contents are calculated by taking a single random sample from each article-segment (sentence): i.e., ignoring multiple codings.

<sup>1</sup> Feature density is defined as the inverse of sparsity:  $\text{density} = \frac{|X_{m,n} \neq 0|}{m \times n}$

<sup>2</sup> Number of observations (rows) in the feature matrix after removing irrelevant objects. The original article-segment feature matrix contained 2,000 objects while the original sentence feature matrix contained 10,000 objects (2,000 times 5).

<sup>3</sup> The variance refers to the variance in the sentence and article-segment feature matrixes used to train the classifier.

<sup>4</sup> The full dataset refers to the set of objects coded, including irrelevant and neutral codings while the classifier dataset excludes all objects coded as irrelevant or neutral. The classifier dataset is used to train the classifier.

<sup>4</sup>We do end up with more sentences than article-segments, thus even if sentences have smaller variances than article-segments the additional number of sentences could compensate, but the ratio is high enough here to suggest that article-segments have the advantage.

## 6 Choosing Between Types of Coders: Low Quality and Low Cost vs. High Quality and High Cost

Who should the analyst have code the data in the training dataset? There are many possible sources of coders, including the analysts themselves. But for large tasks, we can imagine an analyst considering either hiring undergraduate or graduate students or outsourcing the task to a broader labor pool such as those contracted by CrowdFlower or Amazon’s MTurk. The main advantage of using students is that they can be carefully trained as a group. This training likely produces higher quality coding than that of untrained crowd coders by (a) instructing coders to evaluate texts based on the agreed-upon codebook (thus minimizing any effect of individual bias), thereby (b) yielding higher inter-coder reliability, and (c) yielding codings that are more precise (i.e., codings with less variance per object coded)—a point that will be crucial in our discussion below. The main advantage of using crowd coders, however, is that because the labor pool is so large, crowd coders can code the same number of documents much more quickly and, typically, at a much lower cost. Cost and time being equal, we prefer to use high-quality coders. However, even low-quality coders can provide useful information if we aggregate the responses in order to take advantage of the ‘wisdom of the crowds’ (Benoit et al. 2016). The quality of coders also informs how many total codings (i.e., instances of an object being coded) are needed; the higher the quality of coders, the fewer total codings are needed.

How are we to gauge coder quality? Normally analysts evaluate inter-coder reliability, or agreement, among coders. Such measures are useful for evaluating a coding *scheme* but less so for evaluating coder *quality* because high agreement between coders may indicate consistent miscoding.<sup>5</sup> It is important to verify the validity of a coding scheme,

---

<sup>5</sup>See Grimmer, King and Superti for a discussion of the problem of inter-coder reliability, and a novel solution to account for human coder uncertainty, thus reducing bias in the coding estimates (2015). We do not apply their method here, since our aim is to test and report to the reader the relative benefits of making different decisions of which coders to choose. But once the analyst has made a decision, we suggest also using Grimmer et al.’s method for minimizing biased coder estimates.

namely by minimizing coder disagreement, and thus analysts should report inter-coder reliability statistics. But once inter-coder reliability is established, we wish to minimize measurement error in the training dataset. Given a fixed budget, this could be accomplished either by having high-quality coders coding fewer documents or by having low-quality coders coding more documents. In order to assess this tradeoff, we first propose a measure of coder quality—average object-specific coding variance (*AOCV*)—that reflects the amount of measurement error in the mean estimate of tone across coders. We then show how analysts can use this *AOCV* measure, along with the relative cost of two coder pools, to decide which coder pool to use.

We motivate our analysis of coder-choice by considering multiple coders coding a single object. We begin by assuming that each coder produces an unbiased estimate of the object to be coded.<sup>6,7</sup>

Given this assumption of unbiased estimates, any given coding by coder  $j \in J$  about some object  $i \in I$  will be the truth ( $\theta_i$ ) plus some error ( $\epsilon_{ij}$ ). We can then define the ‘quality’ of a coder  $j$  in terms of the average amount of error or the variance in their codings,  $\sigma_j^2$ . Because we have assumed unbiasedness, this variance is the variance about the true tone of the object. Since some objects will be more ambiguous and thus harder to code, they will differ in the amount of error they generate. We can thus refine our

---

<sup>6</sup>We note that this is an assumption of convenience here, not necessity. In comparing coder pools, we would want to minimize mean squared error about the truth. But to simplify things here, we assume the coders are unbiased, and thus compare variance across the groups. Of course, we recognize that all human beings are biased. Compared to the general population, for example, undergraduates tend to be more liberal and from higher socioeconomic families (Krupnikov & Levine 2014), whereas crowd coders tend to be more liberal and of lower socioeconomic status (Levay, Freese & Druckman 2016). Both characteristics could result in biased codings. Thus, analysts should take measures to minimize individual-level bias, such as pilot testing coding instructions and redacting metadata from documents. Most importantly, in order to satisfy the assumption of unbiased estimates, the analyst should assess the degree to which each individual coder is able to conform to the ground truth as established by a ‘gold standard’ subset of coded texts, usually coded by the analysts themselves, that adheres to the measurement intentions of the codebook. In order to minimize bias, coders who do not demonstrate inter-coder reliability with this gold standard sample of text should be excluded. In the analyses we present below, we performed exactly this culling exercise with both the undergraduate and crowd coders.

<sup>7</sup>If we have a coding scheme that is binary, then talking about the variance of coding and maintaining our unbiasedness assumption would be less tenable. With ordinal schemes this assumption seems reasonable. Below we consider two different coding schemes: a 5-point and 9-point scale.

description to say that coder  $j$  has some variance in their coding about each object  $i$ ,  $\sigma_{j,i}^2$ , and for two different objects,  $k$  and  $l$ :  $\sigma_{j,k}^2 \neq \sigma_{j,l}^2, \forall k, l$ . If we assume the errors coders make are independent, then it follows immediately that the variance in our estimate of the truth of a single object drops as we add additional coders for that object.

For simplicity's sake, we assume there are two types of coders: CrowdFlower (CF) coders and undergraduate (UG) coders. We also assume that each CF coder has variance  $\sigma_{cf,i}^2$  and each UG coder has variance  $\sigma_{ug,i}^2$  for object  $i$ . In other words, we assume that all CF coders are of identical quality to one another and all UG coders are of identical quality to one another. Under these assumptions, given the relative cost per coding in each coder pool and the analyst's overall budget constraints, we can select the combination of the type and number of coders most likely to produce a training dataset that minimizes the error for the mean coding of an object ( $Y$ ). Say that, over all of our objects to be coded, the average variance of CF coders is  $\sigma_{cf}^2$ , and the average variance of UG coders is  $\sigma_{ug}^2$ .<sup>8</sup> Assume that  $\sigma_{cf}^2$  is higher than  $\sigma_{ug}^2$ . If we measure the tone of an object  $i$  as the mean coding of that object, the choice between UG coders and CF coders is based on the *relative* variance of the *mean* coding of the object  $i$  by each group. If we have  $J_1$  UG coders, and  $J_2$  CF coders, then the variance of the mean coding over all objects—that is, the average object-specific coder variance (*AOCV*)—will be  $\sigma_{ug}^2/J_1$  for UG coders and  $\sigma_{cf}^2/J_2$  for CF coders. This is the relevant comparison of coder quality, where higher quality coders have smaller variance. We care not about the error from a single coder ( $Y_i - \widehat{Y}_{i,j}$ ). Rather, we care about the error from the *mean* coding of an object: ( $Y_i - \bar{Y}_i$ ).

Given a fixed budget constraint, we need to account both for quality and for cost in our comparison. Say that each UG coding costs  $z$  times as much as each CF coding. Then if we have a fixed budget constraint, implying that  $J_2 = z \times J_1$ , the relevant comparison is between  $AOCV_{ug} = \sigma_{ug}^2/J_1$ , and between a new cost-adjusted calculation for the crowd

---

<sup>8</sup>Note that each object has its own variance, so we can estimate the variance for our coders for each object  $\sigma_{cf,i}^2$  and  $\sigma_{ug,i}^2$  and then average each of those object-specific variances over all objects in our sample to get  $\bar{\sigma}_{cf}^2$  and  $\bar{\sigma}_{ug}^2$ . To keep the notation manageable we use  $\sigma_{cf}^2$  and  $\sigma_{ug}^2$  to represent these averages over objects.

coders,  $AOCV_{cf} = \sigma_{cf}^2 / (z \times J_1)$ . If each group of coders would be coding exactly the same set of objects, and if we are simply choosing based on cost, we pick CF coders if:  $\frac{\sigma_{cf}^2}{\sigma_{ug}^2} < z$ .

In other words, we should choose the CF coders over UG coders if the ratio of the variance of coding of CF coders to the variance of coding of UG coders is less than the ratio of the cost of the two types of coders. In that case the higher number of less precise CF codings will lead to a more precise estimate than the smaller number of more precise UG codings. Thus, this choice is easy to make given estimates on the two variance parameters, obtained from a sample of objects coded by multiple coders from each coding pool. As we describe below, knowing the variance of the coders is crucial to another (joint) the number of objects to code and how many coders to use.

Below we assess the quality of UG coders vis-à-vis CF coders (at least those of each type we used for our data) by applying our *AOCV* measure to a subset of sentences coded by each group and a broader set of sentence codings within each group. We believe trained UG coders will have lower variance than crowd workers. To test this expectation, we calculate the *AOCV* and compare it to a traditional measure of coder quality (average pairwise inter-coder agreement, APIA).

For this analysis, between one and ten undergraduates were asked to code 4,195 unique sentences (Dataset 3SU in Appendix Table 1). Of these, two or more coders coded 1,051 as relevant. A subset of these sentences (3,885) was also coded by 3-6 CrowdFlower workers (Dataset 4SC). At least two coders coded 1,788 of these sentences as relevant. A total of 420 sentences were coded as relevant by at least two coders in each coder pool (Datasets 2SU and 2SC). For each coder pool we compute the variance of the codings for each sentence and then average the object-specific variance. This takes into account the varying number of coders per sentence.<sup>9</sup>

---

<sup>9</sup>Sentences were coded by each group using a 5-category scheme. Codings were recoded to -1 (negative), 0 (mixed, neutral, not sure), and +1 (positive) before computing the variance. This analysis was carried out before we standardized on a 9-point ordinal scale.

Table 11 reports both measures. As expected, UG codings exhibit lower  $AOCV$  and higher average (APIA) than do CF codings. In the set of sentences coded by both groups,  $AOCV_{cf}$  was 0.53 while  $AOCV_{ug}$  was almost half that (0.28). (See Column 1, Rows 3 and 4.) In the population of sentences coded by each group,  $AOCV_{cf}$  was similarly larger (0.57) than  $AOCV_{ug}$  (0.31). (See Column 1, Rows 1 and 2.) The APIA is also higher for UG coders than for CF coders (Column 2). For the overlapping sample of sentences, APIA for UG coders is 0.84 while for CF workers it is 0.70. The comparisons are similar for the full set of sentences coded by each group (0.83 for UG and 0.72 for CF).

**Table 11: Comparison of Inter-coder Reliability of Sentences Coded by Undergraduate Students and by Crowd Workers**

Coders	Variance	APIA	Sentences	Codings
Undergraduate Students (all)	0.31	0.83	1051	3254
Crowd workers (all)	0.57	0.72	1788	4227
Undergraduate Students (overlap)	0.28	0.84	420	1032
Crowd workers (overlap)	0.53	0.70	420	1226

Note: *Variance* is average variance of codings within sentences, *APIA* is the average pairwise inter-coder agreement, *sentences* indicates the total number of unique sentences coded, and *codings* indicates the number of relevant codings. Sentences coded as not relevant are excluded from the analysis. The first two rows correspond to *all* the sentences coded by undergraduate and crowd coders, respectively, Datasets 3SU and 4SC. The second two rows correspond to the subset of sentences coded by *both* students and crowd coders, Datasets 2SU and 2SC.

With this information in hand and still assuming coders are unbiased, we can determine the number of CF coders ( $J^*$ ) needed to get an  $AOCV$  similar to the codings of the  $J_1$  UG coders for a fixed number of objects coded by all  $J_1$  UG coders. We can approximate  $J^*$  as  $AOCV_{cf}/AOCV_{ug} = \frac{0.57}{0.31} = 1.84$ , which implies we need nearly twice as many CF coders per sentence as UG coders to get the same variance. Given an estimate of the relative cost of coding for each type of coder, the analyst can determine the most cost effective way to proceed. In our example, if the relative cost of CF coders is about half (or less) that of UG coders, the analyst would choose the CF coders. However, choosing CF

coders does *not* imply that we should have multiple CF coders code each object, as we explain in the next section. It simply implies that we get more coding effectiveness per dollar spent using CF coders than UG coders.

Ultimately budgets—time and money—constrain coding decisions. We recommend the analyst have each potential coding pool code a small set of objects (ideally by the same number of coders) and then compare the ratio of the *AOCV* in each group with the ratio of the per-coder cost in each group. If the *AOCV* ratio is greater than the cost ratio, the analyst should use larger numbers of the lower quality coders. This calculation does not tell us how many coders to use nor give us any purchase on the number of objects to code in the creation of the training dataset. We turn to these questions next.

### **Choosing Between Types of Coders: Low Cost/Quality vs. High Cost/Quality**

#### **Advantages:**

*Low Cost/Quality Coders:* Less expensive, faster completion.

*High Cost/Quality Coders:* More precise codings (i.e., lower average object-specific coding variance, or *AOCV*).

**Findings:** By calculating the ratio of *AOCV* between the lower cost/quality (Crowd-Flower) coders and the higher cost/quality (undergraduate) coders, we found that if the cost of hiring lower cost/quality coders was less than half as much as hiring higher cost/quality coders, then given a fixed budget we should use the lower cost/quality coders.

**Advice:** Collect a sample of codes from each coding pool and compute the *AOCV* for each. If the ratio of the *AOCV* of lower cost/quality coders to higher cost/quality coders is LESS than the ratio of the cost of lower cost/quality in the two groups, then use the lower cost/quality coders.

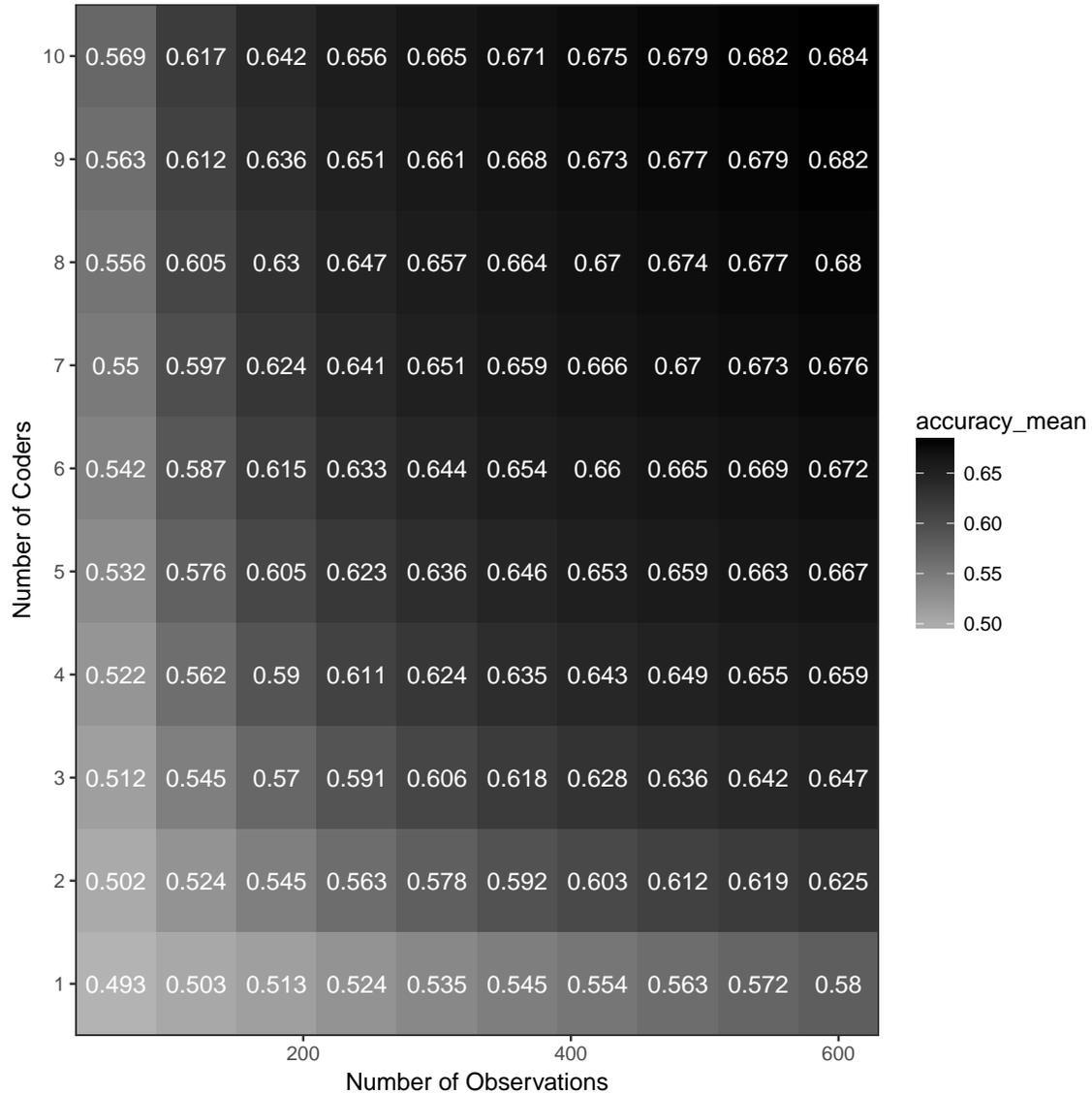
## **7 Allocating Total Codings: More Documents vs More Coders**

The analysis in section 3.2 of the paper can also be used to demonstrate the accuracy for different combinations of number of coders and number of documents. Using the same

approach described in the paper, we(1) generate 20,000 documents that have a true value between 0 and 1 based an underlying linear model using 50 independent variables, (2) convert each to a probability with a logit link function, (3) simulate unbiased coders with  $AOCV = 0.81$  to produce a continuous coding of a subset of documents and (4), we convert each continuous coding to a binary (0/1) classification. Using these codings we estimate an L2 logit where we vary the number of unique documents in the training dataset from  $n = 60$  to  $n = 600$  by increments of 60, and the number of coders per document from  $j = 1$  to  $j = 10$ . For each combination of parameters we drew 15,000 samples of generated data, estimated the model on the training dataset, and then used the estimated parameters to compute accuracy on the test dataset.

In Figure 3 we show the accuracy measured as the percent correctly predicted for different combinations of number of coders and number of documents. The results confirm (a) coding more unique documents,  $n$ , improves predictive accuracy for any number of coders,  $j$ :  $PCP_{n*j} < PCP_{(n+r)*j} \forall r > 0$  and (b) for any given number of unique objects coded,  $n$ , additional coders improves predictive accuracy, although the benefits are smaller than those gained by adding objects.

Figure 3: Heat Map of Accuracy by Number of Coders and Number of Documents



Note: Results are based on simulations described in the text. Shading of cells indicate accuracy of a classifier trained with that specific number of coders and documents.

## 8 Dictionaries vs SML: Accuracy and Precision in UG Truth

Table 12 below presents the top predictive n-grams for the classifier used in the analysis in the paper. Prima facia inspection indicates we are capturing tone when we apply our classifier to this dataset. The top predictive negative n-grams (stemmed) include “declin”, “recess”, “cost”, “unemploy”, “slump”, “deficit”, “plung”, “fear”, “loss”, and “layoff”, words closely associated with a poor economy. Similarly the list of top predictive positive n-grams begins with words we associate with a strong or improving economy: “gain”, “strong”, “rise”, “growth”, “advanc”, “recoveri”. The lists also include less obvious n-grams, namely “washington”, “american”, “school”, “day” (negative n-grams) and “new york”, “januari”, and “person” (positive n-grams). But if these words frequently co-occur in articles with other highly predictive n-grams, these too will contribute to accurate predictions of article tone.

Table 12: Top Predictive N-grams in Classifier

Negative n-grams

---

declin, recess, cost, unemploy, their, slump, off, offic, fell, down, deficit, loss, drop, plung, washington, american, school, day, problem, hous, fear, presid, chief, case, anoth, system, adjust, much, peopl, friday, worst, lend, layoff, part, the most, limit, our, need, be, health

Positive n-grams

---

gain, strong, rise, growth, advanc, recoveri, year, januari, person, sale, earlier, rose, meet, incom, better, save, the fed, expect the, continu, into, improv, gold, manufactur, current, optim, also, activ, new york, increas, while, set, spend, three, york, market, good, two, survey, progress, payrol

Note: Analysis is based on Dataset 5AC, Appendix Table 1. These lists include the n-grams associated with the highest and lowest coefficients, as estimated by our machine learning classifier.

We also assessed accuracy (the percent correctly classified) and precision (the percent of positive articles classified as positive) of each of our classifiers with respect to UG

Truth. The results are presented in Figure 4. The left panel presents the accuracy of the classifiers while the right panel presents their relative precision as in the body of the text. We again include a dotted line in each panel of the figure to represent the percentage of articles in the modal category. These findings parallel those with respect to CF Truth. Specifically, our baseline SML classifier correctly predicted coding by undergraduates in 74.0% of the articles they coded. In comparison, SentiStrength correctly predicted 71.2%, Lexicoder 64.8% and the Hopkins 21-Word Method 58.0% of the articles in UG Truth. With respect to precision, which is the more difficult task here as positive articles are the rare category, the SML classifier even more starkly outperforms the dictionaries. The baseline SML model correctly predicts positive articles 66.7% of the time while SentiStrength does so 50.0% of the time and Lexicoder and Hopkins 21-Word Method do so 37.9% and 16.3% of the time, respectively. In sum, all dictionaries are both less accurate and less likely to identify an article as positive when it was coded as such by humans than our baseline SML model.

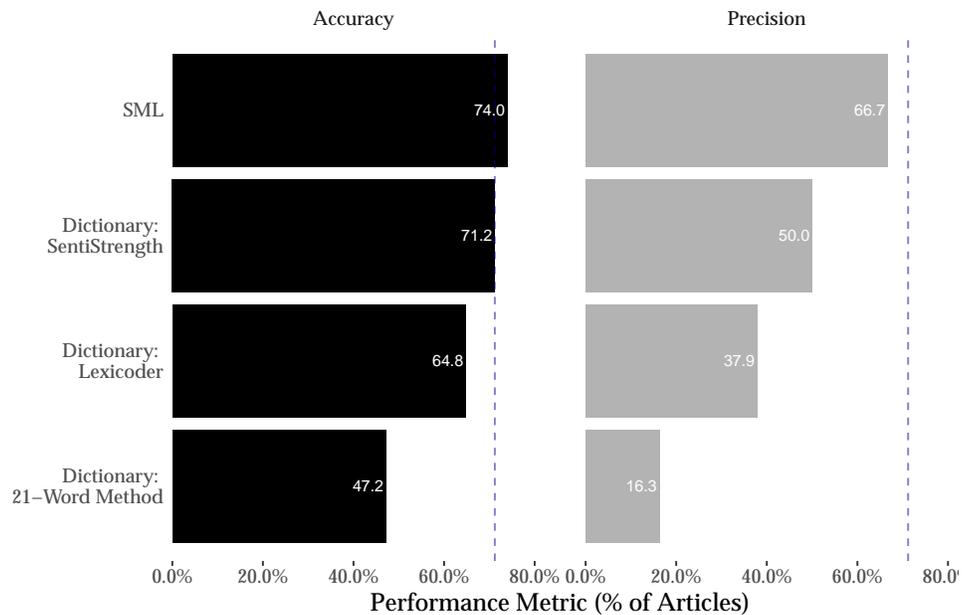
## 9 Convergent Validity

Yet another way analysts evaluate the face validity of classifiers is via convergent validity. The ‘convergent validity’ approach estimates the extent to which two measures that should, in theory, be related are, in fact, empirically related. We expect that media tone of the economy is highly correlated with both real world economic conditions and subjective evaluations of the economy. Thus if our classifiers are producing accurate measures of tone, those measures should also be correlated with these quantities.<sup>10</sup>

---

<sup>10</sup>To create monthly measures of tone, we aggregated the article level scores for both SML classifiers and for SentiStrength and Lexicoder by summing the article level scores multiplied by the number of words per article in a given month and dividing by the total number of words in all articles in that month. This procedure allows longer articles to be given more weight. Monthly scores of the Hopkins et al. 21-Word Method are generated as in Hopkins et al. (2017) by a) calculating the fraction of articles per month mentioning each of 21 economic terms/stems and b) summing these fractions, with positive and negative words having opposite signs.

**Figure 4: Performance of Machine Learning and Dictionary Methods—Accuracy and Precision**

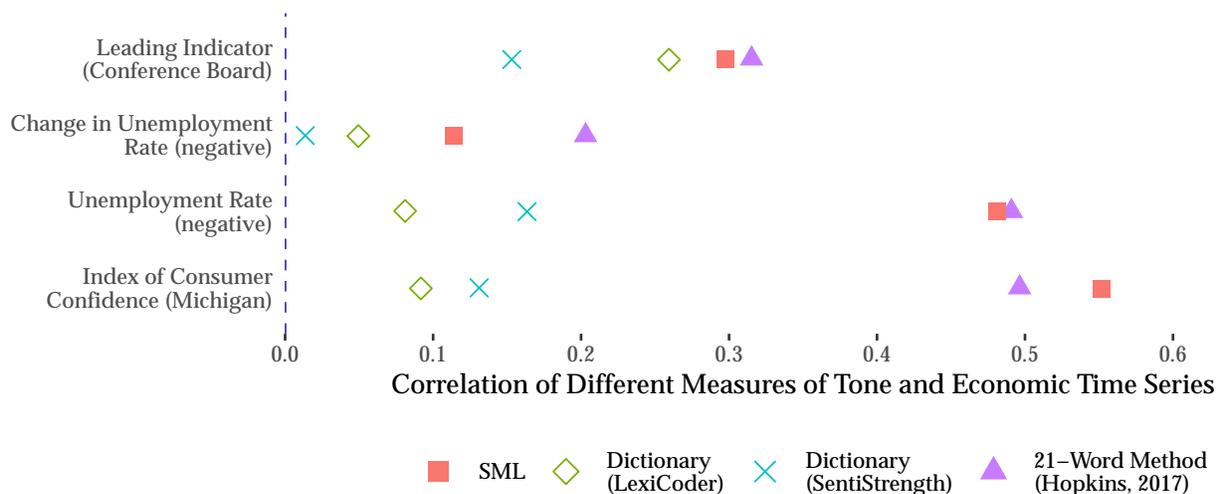


Note: Accuracy (percentage of articles correctly classified) and precision (percentage of positive articles predicted to be positive) are reported for the ground truth dataset coded by undergraduate students (left) and by 10 CrowdFlower coders (right). The dashed vertical lines indicate the baseline level of accuracy if the modal category is always predicted. The corpus used in the analysis is based on the keyword search of *The New York Times* 1980-2011 (see the text for details).

In Figure 5 we report the correlation of tone produced by each of the dictionary measures and our SML baseline classifier (the baseline SML classifier and the SML classifier with directional terms are perfectly correlated) with each of: a) the unemployment rate (negative), b) changes in the unemployment rate (negative), c) the Conference Board’s leading economic indicator index, and d) the University of Michigan Index of Consumer Confidence. The series generated by the article-level classifier consistently correlated more highly with each indicator than did SentiStrength and Lexicoder. Hopkins 21-Word Method, which was optimized to maximize these correlations, produced a series more highly correlated with changes in the unemployment rate than any of the other classifiers. In particular, tone, as measured by the baseline machine learning classifier, corre-

lated most strongly with the ICS (0.56). Hopkins 21-Word Method generated a correlation with the ICS that is almost as large (0.50), while Lexicoder and SentiStrength were only mildly correlated with the ICS (0.10 and 0.13, respectively). The SML classifier and Hopkins 21-Word Method produced the same correlation with the unemployment rate (-0.49) and nearly the same correlation with the leading indicator index (0.30 and 0.32, respectively). In contrast, SentiStrength and Lexicoder correlated at -0.17 and -0.08 with the unemployment rate and 0.15 and 0.26, respectively, with the leading indicator index. Finally, Hopkins-21 Word method was the most strongly related to changes in the unemployment rate (-0.21) with the SML classifiers correlated at -0.11 and SentiStrength and Lexicoder at -0.02 and -0.05, respectively. In general, the dictionary methods did not produce measures of tone that maximize convergent validity. Hopkins 21-Word Method is an exception, but, as we noted, it was optimized to maximize these correlations.

Figure 5: Performance of Machine Learning Classifier and Dictionary Methods—Convergent Validity



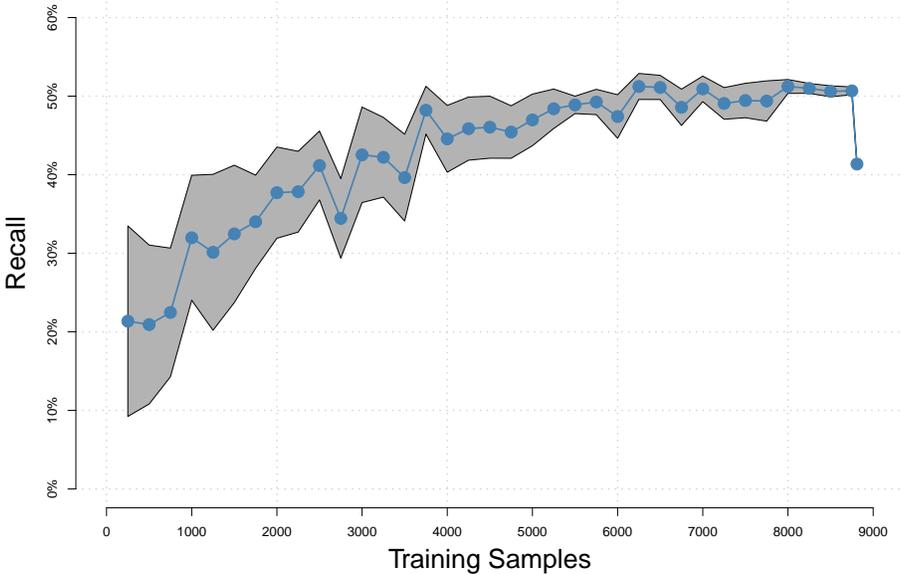
Note: The period of analysis is 1948–2014 (unemployment), 1959–2014 (leading indicator), 1978–2014 (index of consumer confidence). Points are jittered and correlations involving unemployment or change in unemployment are multiplied by -1 for visual clarity. The corpus used in the analysis is based on the keyword search of *The New York Times* 1980–2011 (see the text for details).

## 10 SML Classifier Performance as a Function of the Size of the Training Dataset

In Section 4 of the paper we analyzed the accuracy of our baseline SML classifier as a function of the size of the training dataset. Here we present our findings for recall (Table 6) and precision (Table 7). Recall that we drew 10 random samples of 250 articles each from the full training dataset. Using the same method as discussed in the text, we estimated the parameters of our SML classifier on each of these 10 samples. We then used each of these estimates of our classifier to predict the tone of articles in CF Truth, recording accuracy, precision, and recall for each replication. We repeated this process for sample sizes of 250 to 8,750 by increments of 250. The x-axis gives the size of the training data set and y-axis reports the average recall or precision in CF Truth for the given sample size. The figures include a 95% confidence interval for all subsets of the complete training dataset.

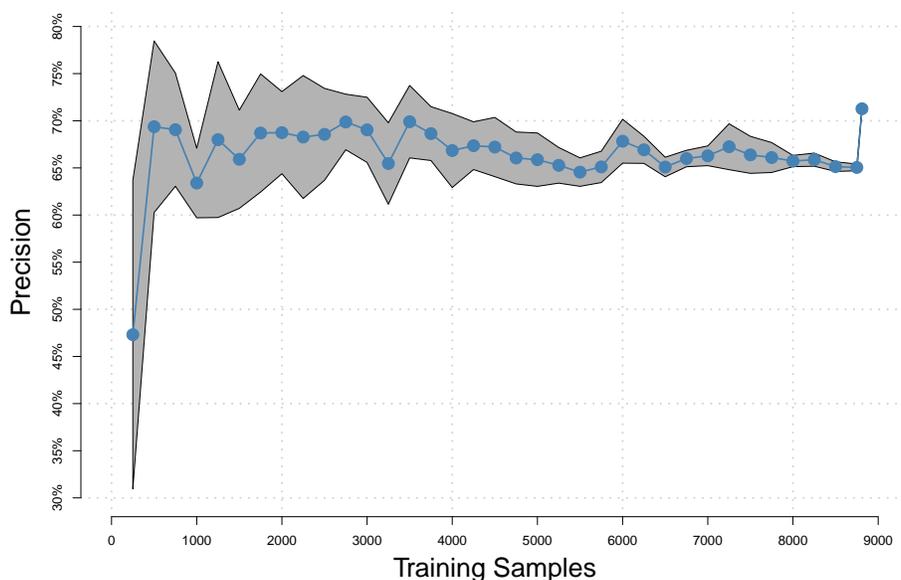
As the size of the training dataset increases, recall—the fraction of articles identified as positive that were positive—improves consistently from about 20% when the sample training dataset contains 250 articles to about 50% for training datasets containing 4,000 articles. Further increases in the size of the training dataset appear to have no effect. Precision—the fraction of items classified as positive that really are positive, however, is quite low (about 47%) for  $N = 250$  but jumps up and remains relatively flat between 65 and 70% for all sized training datasets 500 and greater.

Figure 6: Recall of the Baseline Machine Learning Classifier as a Function of Size of the Training Dataset



Note: We drew 10 random samples of 250 articles each from the full training dataset (dataset 6AC, Table 1) of 8,750 unique codings of 4,400 unique articles (three to five crowd coders labeled each article) in the *New York Times* randomly sampled from the years 1947 to 2014. Using the same method as discussed in the text, we estimated the parameters of our SML classifier on each of these 10 samples. We then used each of these estimates of our classifier to predict the tone of articles in CF Truth. We repeated this process for sample sizes of 250 to 8,750 by increments of 250, recording the proportion of articles classified as positive that were coded as positive by the crowd workers.

Figure 7: Precision of the Baseline Machine Learning Classifier as a Function of Size of the Training Dataset



Note: We drew 10 random samples of 250 articles each from the full training dataset (dataset 6AC, Table 1) of 8,750 unique codings of 4,400 unique articles (three to five crowd coders labeled each article) in the *New York Times* randomly sampled from the years 1947 to 2014. Using the same method as discussed in the text, we estimated the parameters of our SML classifier on each of these 10 samples. We then used each of these estimates of our classifier to predict the tone of articles in CF Truth. We repeated this process for sample sizes of 250 to 8,750 by increments of 250, recording the proportion of articles classified as positive that were coded as positive by the crowd workers.

## References

- Benoit, Kenneth, Drew Conway, Benjamin E Lauderdale, Michael Laver & Slava Mikhaylov. 2016. "Crowd-sourced text analysis: Reproducible and agile production of political data." *American Political Science Review* 110(2):278–295.
- Grimmer, Justin, Gary King & Chiara Superti. 2015. "The Unreliability of Measures of Intercoder Reliability, and What to do About it." Working paper.
- Hopkins, Daniel J, Eunji Kim & Soojong Kim. 2017. "Does newspaper coverage influence or reflect public perceptions of the economy?" *Research & Politics* 4(4):2053168017737900.

Krupnikov, Yanna & Adam Seth Levine. 2014. "Cross-sample comparisons and external validity." *Journal of Experimental Political Science* 1(1):59–80.

Levay, Kevin E, Jeremy Freese & James N Druckman. 2016. "The Demographic and Political Composition of Mechanical Turk Samples." *Sage Open* 6(1):1–17xs.