Automated Text Classification of News Articles: A Practical Guide.[*]

Pablo Barberá [†] **Amber E. Boydstun**[‡] **Suzanna Linn**[§]

Jonathan Nagler[¶] **Ryan McMahon**[||]

August 2, 2019

Abstract

Automated text analysis methods have made possible the classification of large corpora of text by measures such as topic and tone. Here, we provide a guide to help researchers navigate the consequential decisions they need to make before any measure can be produced from the text. We consider, both theoretically and empirically, the effects of such choices using as a running example efforts to measure the tone of *New York Times* coverage of the economy. We show that two reasonable approaches to corpus selection yield radically different corpora and we advocate for the use of keyword searches rather than pre-defined subject categories provided by news archives. We demonstrate the benefits of coding using article-segments instead of sentences as units of analysis. We show that, given a fixed number of codings, it is better to increase the number of unique documents coded rather than the number of coders for each document. Finally, we find that supervised machine learning algorithms outperform dictionaries on a number of criteria. Overall, we intend this guide to serve as a reminder to analysts that thoughtfulness and human validation are key to text-as-data methods, particularly in an age when it is all-too-easy to computationally classify texts without attending to the methodological choices therein.

[*] This material is based in part on work supported by the National Science Foundation under IGERT Grant DGE-1144860, Big Data Social Science, and funding provided by the Moore and Sloan Foundations. Nagler's work was supported in part by a grant from the INSPIRE program of the National Science Foundation (Award SES-1248077).

[†] Pablo Barberá is an Assistant Professor of International Relations at the University of Southern California. He can be reached at pbarbera@usc.edu

[‡] Amber Boydstun is an Associate Professor of Political Science at the Department of Political Science at the University of California, Davis.

[§] Suzanna Linn is a Professor of Political Science at Penn State University.

[¶] Jonathan Nagler is a Professor of Politics at New York University and a co-Director of the NYU Social Media and Political Participation Lab.

[||] Ryan McMahon is a Data Scientist at Verisk Maplecroft.

1 Introduction

The analysis of text is central to a large and growing number of research questions in the social sciences (Grimmer & Stewart 2013). While analysts have long been interested in the tone and content of such things as media coverage of the economy (Soroka, Stecula & Wlezien 2015, Young & Soroka 2012, Goidel et al. 2010, De Boef & Kellstedt 2004, Tetlock 2007, Doms & Morin 2004), congressional bills (Jurka et al. 2013, Hillard, Purpura & Wilkerson 2008), party platforms (Grimmer, Messing & Westwood 2012, Monroe, Colaresi & Quinn 2008, Laver, Benoit & Garry 2003), and presidential campaigns (Eshbaugh-Soha 2010), the advent of automated text classification methods combined with the broad reach of digital text archives have led to an explosion in the extent and scope of textual analysis. Whereas researchers were once limited to analyses based on text that was read and hand-coded by humans, machine coding by dictionaries and supervised machine learning tools are now the norm (Grimmer & Stewart 2013). The time and cost of the analysis of text has thus dropped precipitously. But the use of automated methods for text analysis requires the analyst to make multiple decisions that are often given little consideration yet that have consequences that are neither obvious nor benign.

Before proceeding to classify documents, the analyst must: (1) select a corpus; and (2) choose whether to use a dictionary method or a machine learning method to classify each document in the corpus. If a supervised machine learning (SML) method is selected, the analyst must also: (3) decide how to produce the training dataset—select the unit of analysis, the number of objects (i.e., documents or units of text) to code, and the number of coders to assign to each object.

In each section below, we first identify the options open to the analyst, and present the theoretical trade-offs associated with each option. Second, we offer empirical evidence illustrating the degree to which these decisions matter for our ability to predict the tone of coverage of the U.S. national economy in the *New York Times*, as perceived by human readers. Third, based on the theoretical discussion and empirical analyses, we provide recommendations to the analyst on how to best evaluate their choices. Throughout, our goal is to provide a guide for analysts facing these decisions in their own work.

Because our discussion of SML and dictionaries necessarily relies on having produced a training dataset, we present the results in the following order. First we discuss selection of the corpus, then we discuss issues in creating a training dataset, then we compare the benefits and trade-offs associated with choosing to use SML versus dictionary methods.

The discussion in this paper is limited to coding the *tone* of text, rather than other variables such as topics or events.[1] But much of what we present is applicable to the analysis of text more broadly, both when using a computational approach and even (in the first stage we discuss below) when using manual content analysis.

Some of the results and advice we present here may seem self-evident. If one chooses the wrong corpus of media to code, for example, it is intuitive that no coding scheme will accurately capture the 'truth' of how humans would perceive what the media is presenting. We also show that less obvious decisions matter. And that they matter a lot. We show that two reasonable attempts to select a corpus from the same media source can produce dramatically different outcomes on a number of dimensions. In our running example, using keyword searches produces a larger corpus than using pre-defined subject categories (developed by LexisNexis), with a higher proportion of relevant articles. Since keywords also offer the advantage of transparency over using subject categories, we conclude below that keywords are to be preferred over subject categories in choosing a corpus. We then offer guidance on how to produce a training dataset if the analyst will be using SML, focusing on three decisions. First, we show that it makes surprisingly little difference whether the analyst codes text at the sentence level or the article-segment level,

[1] Tone is also referred to as 'sentiment' in the computer science literature.

suggesting that taking advantage of the efficiency of coding at the segment level is to be preferred in practice to coding at the sentence level. Second, we show that maximizing the number of objects coded rather than having fewer objects each coded by more coders provides the most efficient means to optimize the performance of SML methods. Finally, we demonstrate that SML out-performs dictionary methods on a number of different criteria, including accuracy and precision, and thus we conclude that the use of SML is to be preferred to the use of dictionaries, provided the analyst is able to produce a training dataset of sufficiently high quality and quantity.

Before proceeding to describe the decisions at hand, we note two key assumptions associated with measuring tone. First is the assumption that the analyst's goal is to produce a measure of tone that accurately represents the tone of the text as read by humans.[2] Second is the assumption that, on average, the tone of a given text is interpreted by all people in the same way; in other words, that there is a single, 'true' tone inherent in the text that has merely to be extracted. Of course, this second assumption is the harder one to maintain. In the case of the tone of media coverage of the economy, different readers of the same text may interpret the story differently, for instance because of different levels of economic sophistication or different economic interests. However, we are comfortable proceeding with a slightly relaxed version of the assumption that all respondents perceive the same truth - rather we proceed assuming that while different individuals may perceive a different value of the same object, there is a grand mean of what all individuals would perceive. Then, we rely on the extensive literature on the concept of the wisdom of the crowds—the idea that aggregating multiple independent judgments about a particular question can lead to the correct answer, even if those individual assessments are coming from individuals with low levels of information.[3] As Benoit (2016) demonstrated,

[2] In other words, we assume that the analyst is less interested in capturing the tone of the text that was intended by its author, and more interested in capturing how the public, in general, would interpret the tone. This assumption may not always hold. In the context of a social media study, for example, an analyst might be more interested in the intended tone of a Tweet than in the tone as it comes across to a general audience. But in general we think the assumption applies to most analysts using text as data.

[3] The core notion of the 'wisdom of the crowds,' first introduced by Condorcet in his jury theorem (1972),

this logic can also be applied to the coding of political text in order to obtain correct inferences about its content. Thus we will proceed assuming that: (1) the analyst is most interested in capturing human interpretation of the tone, and (2) we can approximate the 'true' tone of a newspaper article, or the mean of what all individuals would believe it to be, by aggregating multiple independent codings of it. Below we describe each decision the analyst must make in proceeding.

2 Selecting the Corpus: Keywords vs. Subject Categories

The first decision confronting the analyst of text is how to select the corpus. The analyst must first define the **universe**, or source, of text. The universe may be well-defined and relatively small, e.g., the written record of all legislative speech in Canada over the last decade, or it may be broad and large in scope, e.g., the set of all text produced by the 'media.' Having identified the relevant universe, the analyst must define the **population**, *the set of objects (e.g., articles, bills, tweets) in the universe relevant to the analysis*. The population may correspond to the universe, but often the analyst will be interested in a subset of documents in the universe, such as those on a particular topic. In the final step, the analyst selects the **set** of documents that defines the corpus to be classified.

Note the distinction between *defining* the population of interest, and actually identifying it. For instance, the analyst can state their intent is to measure the tone of *all* articles in the *New York Times* relevant to the economy, but whether any article is in that set is not known a priori. Thus the analyst is bound to work with a corpus that both *includes* articles that are not actually in the population of interest, and that *excludes* articles that are in the population of interest.[4]

The challenge is to adopt a sampling strategy that produces a corpus that mimics

is now widely applied in social science (Lyon & Pacuit 2013, Page 2008, Surowiecki 2005, Wagner et al. 2010).

[4] In some cases the corpus may correspond to the entire population, but in our running example, as in any example based on media sources, they will be distinct.

the population. We want to include all *relevant* objects (i.e., minimize false negatives) and exclude any *irrelevant* objects (i.e., minimize false positives). Including irrelevant texts adds noise to measures produced from the sample and adds cost to the production of a training dataset (for analysts who will be using SML). In contrast, a strategy that excludes relevant texts at best produces a noisier measure by increasing sampling variation in any measure produced from the sample corpus.

In addition to the concern about relevance is a concern about representation. With every decision about which words or terms to include or omit from a keyword search, we run the risk of introducing bias. For example, we might find that an expanded set of keywords yields a larger and highly relevant corpus, but if the added keywords are disproportionately negatively toned, or disproportionately related to one aspect of the economy compared to another vis a vis the population, then this highly relevant corpus would be of lower quality. The vagaries of language make this a real possibility. However, careful application of keyword expansion can minimize the potential for this type of error. In short, the analyst should strive for a keyword search that maximizes both relevance and representation vis a vis the population of interest.

One of two sampling strategies is typically used to select the corpus of text for analysis. In the first strategy the analyst selects texts based on subject categories produced by some entity that has already categorized the documents (e.g., by topic). In the case of coding newspaper articles, for example, the media monitoring site LexisNexis has developed an extensive set of hierarchical topic categories, and the media data provider ProQuest offers a long list of fine-grained topic categories, each identified through a combination of human discretion and machine learning.[5]

In the second strategy, the analyst relies on a boolean regular expression search

[5] As another example, the Policy Agendas Project <www.comparativeagendas.net> offers an extensive database of items (e.g., legislative texts, executive texts) that other scholars have already categorized based on an institutionalized list of codes that divide policy areas by topic (e.g., macroeconomics) and sub-topic (e.g., interest rates).

using keywords (or key terms). Typically the analyst generates a list of keywords, based on their subject expertise, which are expected to distinguish between articles relevant to the topic compared to irrelevant articles. For example, in looking for articles about the economy, the analyst would likely choose “unemployment” as a keyword. There is a burden on the analyst to choose terms that can capture documents from across the distribution of what is being studied. An analyst looking to examine articles to measure tone of the economy who started with a keyword set including “recession”, “depression”, and “layoffs” but omitting “boom” and “expansion” runs the risk of producing a biased set of documents. But once the analyst chooses a small set of core keywords, there are established algorithms an analyst can use to move to a larger set.[6] Most such algorithms rely on co-occurrence: if a term co-occurs with “unemployment”, but does not occur too frequently without it, then it is likely to also be about the economy. This additional term would then be added to the set of keywords.

What are the relative advantages of these two sampling strategies? We might expect corpora selected using subject categories defined at least in part by humans to be relatively more likely than keyword-generated samples to capture relevant documents and omit irrelevant documents precisely because humans were involved in their creation. If humans categorize text synchronous with its production, it may also be that category labels account for differences in vocabulary specific to any given point in time. However, if subject categories relies on human coders, changing coders could cause a change in content independent of actual content, and this drift would be invisible to the analyst. More significantly, often, and specifically in the case of text categorized by media providers such as LexisNexis and ProQuest, the means of assigning individual objects to the subject categories provided by the archive (or even by an original news source) are proprietary.

[6] See King, Lam, and Roberts (2016) and citations therein for discussion of keyword generation methods. Note that these methods must start with a human selection of keywords to seed the algorithm, meaning that there is no escaping the need for vigilance in thinking about which keywords are both relevant and representative.

The resulting absence of transparency is a huge problem for scientific research, even if the category is highly accurate (a point on which there is no evidence). Further, as a direct result, the search is impossible to replicate in other contexts, whether across publications or countries. And it likewise can make extension (e.g., the updating of a dataset) impossible. Finally, the categorization rules used by the archiver may change over time. In the case of LexisNexis and ProQuest, not only do the rules used change, but even the list of available subject categories changes over time. As of 2019, LexisNexis no longer even provides a full list of subject categories.[7]

The second strategy, using a keyword search, gives the analyst control over the breadth of the search. In this case, the search is easily transported to and replicable across alternative or additional universes of documents. Of course, if the analyst chooses to do a keyword search, the choice of keywords becomes 'key'. There are many reasons any keyword search can be problematic: relevant terms can change over time, different publications can use overlooked synonyms, and so on.[8]

Here we compare the results produced by using these two strategies to generate corpora of newspaper articles from our pre-defined universe (*The New York Times*), intended to measure the tone of news coverage of the U.S. national economy.[9] As an example of the first strategy, Soroka et al. (2015) selected a corpus of texts from the universe of the *New York Times* from 1980-2011 based on media-provided subject categories using Lexis-

[7] LexisNexis now states that "Due to proprietary reasons, we aren't allowed to share this information [the list of subject categories]." Correspondence with authors, May 30, 2019.

[8] It is important to note that because the underlying set of archived articles can vary over time, based on the media provider's contracts with news outlets and the provider's internal archiving parameters, even the same keyword search performed at two points in time may yield maddeningly different results, although the differences should be less than those suffered using proprietary subject categories (Fan, Geddes & Flory 2013).

[9] A number of analysts have coded the tone of news coverage of the U.S. national economy. The universe of text defined in this body of work varies widely from headlines or front-page stories in the *New York Times* (Blood & Phillips 1997, Wu et al. 2002, Fogarty 2005, Goidel & Langley 1995) to multiple newspapers (Soroka, Stecula & Wlezien 2015), to as many as 30 newspapers (Doms & Morin 2004). Blood and Phillips (1997) coded the full universe of text while others used subject categories and/or keyword searches to produce a sample of stories from the population of articles about the economy.

Nexis.[10] As an example of the second strategy, we used a keyword search of the *New York Times* covering the same time period using ProQuest.[11] We compare the relative size of the two corpora, their overlap, the proportion of relevant articles in each, and the resulting measures of tone of the U.S. national economy produced by each. On the face of it, there is little reason to claim that one strategy will necessarily be better at reproducing the population of articles about the U.S. economy from the *New York Times* and thus a better measure of tone. The two strategies have the same goal, and one would hope they would produce similar corpora.

The subject category search listed by Soroka et al. captured articles indexed in at least one of the following LexisNexis defined sub-categories of the subject "Economic Conditions": Deflation, Economic Decline, Economic Depression, Economic Growth, Economic Recovery, Inflation, or Recession. They also captured articles in the following LexisNexis sub-categories of the subject "Economic Indicators": Average Earnings, Consumer Credit, ConsumerPrices, Consumer Spending, Employment Rates, Existing Home Sales, Money Supply, New Home Sales, Productivity, Retail Trade Figures, Unemployment Rates, or Wholesale Prices. Articles were kept if they had a relevance score of 85 or higher, as defined by LexisNexis, for any of the sub-categories listed above. Post-collection, Soroka et al. manually removed articles not focused solely on the U.S. domestic economy, irrelevant to the domestic economy, shorter than 100 words, or "just long lists of reported economic figures and indicators," (Soroka, Stecula & Wlezien 2015, 461-462). This corpus is the basis of the comparisons below.

To generate a sample of economic news stories using a keyword search, we downloaded all articles from the *New York Times* archived in ProQuest with **any** of the following terms: employment, unemployment, inflation, consumer price index, GDP, gross domes-

[10] Soroka and colleagues generously agreed to share their dataset with us, for which we are deeply grateful, allowing us to perform many of the comparisons in this article.

[11] We used ProQuest because LexisNexis does not have historical coverage for the *New York Times* earlier than 1980, and we wanted to base some of our analyses below on a longer timespan.

tic product, interest rates, household income, per capita income, stock market, federal reserve, consumer sentiment, recession, economic crisis, economic recovery, globalization, outsourcing, trade deficit, consumer spending, full employment, average wage, federal deficit, budget deficit, gas price, price of gas, deflation, existing home sales, new home sales, productivity, retail trade figures, wholesale prices **AND** United States.[12,13,14] We used a filter to remove any articles that mentioned a country name, country capital, nationality or continent name that did NOT also mention U.S., U.S.A. or United States in the headline or first 1,000 characters of the article (Schrodt 2011).

We first compare the number of articles in each corpus (Columns 1 and 2, Table 1). Overall the keyword search produced a corpus containing nearly twice as many articles as did the corpus produced using the subject categories (30,787 vs. 18,895). 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 than did the keyword corpus. Notably, there is a downward trend in the number of articles in the keyword corpus that is not apparent in the subject category corpus.

[12] We obtained articles from two sources: the ProQuest Historical *New York Times* Archive and the ProQuest Newsstand Database. Articles in the first database span the 1947-2010 period and are only available in PDF format and thus had to be converted to plain text using OCR (optical character recognition) software. Articles for the 1980-2014 period are available in plain text through ProQuest Newsstand. We used machine learning techniques to match articles in both datasets and to delete duplicated articles, keeping the version available in full text through ProQuest Newsstand.

[13] We could also have generated keywords using a(n) (un)supervised method or query expansion (King, Lam & Roberts 2016, Xu & Croft 1996, Rocchio 1971, Schütze & Pedersen 1994, Bai et al. 2005, Mitra, Singhal & Buckley 1998). However, those methods are difficult to implement because they generally require unfettered access to the entire population of documents, which we lacked in our case due to access limitations imposed by ProQuest.

[14] Note that, although in theory LexisNexis and ProQuest should have identical archives of *New York Times* articles for the overlapping years, one or both archives might have idiosyncrasies that contribute to some portion of the differences between the keyword-based and subject category-based corpora presented below. Indeed, the vagaries of LexisNexis alone are well documented (Fan, Geddes & Flory 2013, 106).

Table 1: Comparing the Subject Category Corpus with the Keyword Corpus: Total, Unique, and Overlapping (Common Corpus) Article Counts from the *New York Times*, 1980-2011

	Keyword	Subject Category	Common	Unique	Unique
Year	Corpus	Corpus	Corpus	Keyword	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 15 for a description of the methods used to calculate article overlap.

Next we identified the overlap in articles in the two corpora in order to determine how distinctive they are.[15] The third column of Table 1 presents the total number of articles that appear in both corpora (common), while the remaining columns provide the number of articles unique to the keyword corpus (Column 4) and unique to the subject category corpus (Column 5). Overall only 13.9% of the articles in the keyword corpus are included in the subject category corpus and only 22.7% of the articles in the subject category corpus are found in the keyword corpus. In other words, if we were to code the tone of media coverage based on the keyword corpus we would omit 77.3% of the articles in the subject category corpus, while if we relied on the subject category corpus, we would omit 86.1% of the articles in the keyword corpus. There is, in short, shockingly little article overlap between two corpora produced using reasonable strategies designed to capture the same population: the set of *New York Times* articles relevant to the state of the U.S. economy. This finding of course brings up the troubling possibility that even if identical classification mechanisms were applied to them, the two corpora could produce substantially different estimates of the tone of media coverage of the economy.

Having more articles does not necessarily indicate that one corpora is better than the other. The lack of overlap may indicate the subject-category search is too narrow and/or the keyword search is too broad. Perhaps the use of subject categories eliminates articles that provide no information about the state of the U.S. national economy, despite

[15] 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% 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.

containing terms used in the keyword search. In order to assess these possibilities, we recruited coders through the online crowd-coding platform CrowdFlower (now called Figure Eight), who coded the relevance of: (1) 1,000 randomly selected articles unique to the subject category corpus; (2) 1,000 randomly selected articles unique to the keyword corpus, and (3) 1,000 randomly selected articles in both corpora. We present the results in Table 2.[16]

Table 2: Proportion of Relevant Articles by Corpus

Relevance	Articles in both Corpora	Articles Unique to the Keyword Corpus	Articles Unique to the Subject Category Corpus
Relevant	0.44	0.42	0.37
Not Relevant	0.56	0.58	0.63

Note: Cell entries indicate the proportion of articles 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 by three CrowdFlower workers located in the US. Each coder was assigned a weight based on her overall performance before computing the proportion of articles deemed relevant. If two out of three (weighted) coders concluded an article was relevant, the aggregate response is coded as "relevant".

The proportion of relevant articles is given in the top row of Table 2. Overall, both search strategies yield a sample with a large proportion of irrelevant articles, suggesting the searches are too broad.[17] Unsurprisingly the proportion of relevant articles was highest, 0.44, in articles that appeared in *both* the subject category and keyword corpora. Nearly the same proportion of articles unique to the keyword corpus was coded as rele-

[16] Relevance codings were based on coders' assessment of relevance upon reading the first 5 sentences of the article. See Section 1 of the Appendix for the coding instrument. All three coders coded each article (based on its first five sentences), producing 9,000 total codings. Each coder was assigned a weight based on his/her overall performance (the level of the coder's agreement with that of other coders) before computing the proportion of articles deemed relevant. If two out of three (weighted) coders concluded an article was relevant, the aggregate response is coded as "relevant". This is de facto a majority rule criterion as coder weights were such that a single heavily-weighted coder did not overrule the decisions of two coders when there was disagreement. The coding-level proportions were qualitatively equivalent and are presented in Table 2 in Section 3 of the Appendix.

[17] Note again, however, that we only had coders read the first five sentences of each article. It may be that some (or even many) of the articles deemed irrelevant contained relevant information after the first five sentences.

vant (0.42), while the proportion of articles unique to the subject category corpus coded relevant dropped by 13.5%, to 0.37. This suggests the LexisNexis subject categories do not provide any assurance an article provides "information about the state of the economy." Because the set of relevant articles in each corpus is really a sample of the population of articles about the economy and since we want to estimate the population values, we prefer a larger to a smaller sample, all else being equal. In this case, the subject category corpus has 7,291 relevant articles, while the keyword corpus has 13,016 relevant articles.[18] Thus the keyword dataset would give us on average 34 relevant articles per month with which to estimate tone, while the subject category dataset would only give us on average 19 relevant articles per month. Further, the keyword dataset is not providing more observations at a cost of higher noise: the proportion of irrelevant articles in the keyword corpus is lower than the proportion of irrelevant articles in the subject category corpus.

These comparisons demonstrate that the given keyword and subject category searches produced highly distinct corpora and that both corpora contained large portions of irrelevant articles. Do these differences matter? The highly unique content of each corpus suggests the potential for bias in both measures of tone. And the large proportion of irrelevant articles suggests measures of tone produced from each corpus will contain measurement error. But given that we do not know the true tone as captured by a corpus that includes all relevant articles and excludes all irrelevant articles (i.e., in the population of articles on the U.S. national economy), we cannot address these concerns directly.[19] We can, however, determine how much the differences between the two corpora affect the estimated measures of tone. Applying Lexicoder, the dictionary used by Soroka et al. (2012),

[18] The subject category corpus contains 4,290 articles in common with the keyword corpus, of which 44% are relevant, and 14,605 articles unique to the subject category corpus, of which 37% are relevant. Of the 26,497 articles unique to the keyword corpus, 42% are relevant.

[19] As we discuss in Section 3 of the Appendix, an analyst could train the classifier for relevance and then omit articles classified as irrelevant. However, we found that it was difficult to train an accurate relevance classifier, which meant that using it as a filter could lead to sampling bias in the resulting final sample. Since our tests did not show a large difference in the estimates of tone from the sample, we opted for the one-stage classifier as it was a more parsimonious choice.

to both corpora we find a correlation of 0.48 between the two monthly series while application of our supervised machine learning algorithm resulted in a correlation of 0.59.[20] Longitudinal changes in tone are often the quantity of interest and the correlations of changes in tone are much lower, 0.19. and 0.36 using Lexicoder and SML, respectively. These low correlations are due in part to measurement error in each series, but these are disturbingly low correlations for two series designed to measure exactly the same thing. Our analysis suggests that regardless of whether one uses a dictionary method or a supervised machine learning method, the resulting estimates of tone may vary significantly depending on the method used for choosing the corpus in the first place.

The extent to which our findings generalize is unclear—keyword searches may be ineffective and subject categorization may be quite good in other cases. However, keyword searches are within the analyst's control, transparent, reproducible, and portable. Subject category searches are not. We thus recommend analysts use keyword searches rather than subject categories, but that they do so with great care. Whether using a manual approach to keyword generation[21] or a computational query expansion approach[22] it is critical that the analyst pay attention to selecting keywords that are both *relevant* to the population of interest and *representative* of the population of interest. For relevance, analysts can follow a simple algorithm: (a) do a narrow keyword search; (b) do a broad keyword search; (c) code a sample of each corpus for relevance. If (b) returns more objects than (a), **but** the proportion of relevant texts decreases substantially, then use the narrower keyword search. If (b) returns more objects than (a), **and** relevance does not decrease, use (b) (and continue to expand the keyword search until doing so does not

[20] Lexicoder tone scores for documents are calculated by taking the number of positive minus the number of negative terms over the total number of terms (Eshbaugh-Soha 2010, Soroka, Stecula & Wlezien 2015). We use our baseline SML algorithm which we define in more detail later on, but which is based on logistic regression with an L2 penalty. The monthly tone estimates for both measures are the simple averages across the articles in a given month.

[21] For an illustration of good practices in iterative keyword selection, see Atkinson, Lovett and Baumgartner (2014, 379-380).

[22] For methods of query expansion, see King, Lam and Roberts (2016), Xu and Croft (1996), Rochio (1971), Schutze (1994), Bai (2005), and Mitra (1998).

increase the proportion of irrelevant texts). However, the analyst is always going to face the risk of introducing bias by omitting in a non-random way relevant articles. Thus, we recommend analysts utilize established keyword expansion methods but also domain expertise (good old-fashioned subject-area research) so as to expand the keyword selection in a way that does not skew the sample toward an unrepresentative portion of the population of interest. There is potentially a large payoff to this simple use of human intervention early on.

Selecting the Corpus: Keywords vs. Subject Categories

Advantages:
Keywords: Transparent, portable, controlled by the analyst.
Subject Categories: Off-the-shelf, may incorporate human domain expertise.

Findings: In our comparison, these two approaches yield dramatically different corpora, with the keyword search producing a larger corpus with a higher proportion of relevant articles.

Advice: Use keyword searches, following an iterative vetting process to evaluate trade-offs between broader vs. narrower sets of keywords. Avoid subject categories, as their black-box nature can make replication and extension impossible.

3 Creating a Training Dataset: Two Crucial Decisions

Once the analyst selects a corpus, there are two fundamental options for coding sentiment (beyond traditional manual content analysis): dictionary methods and supervised machine learning methods. Before comparing these approaches, we consider the decisions the analyst must make to carry out a necessary step for applying supervised machine learning methods: producing a training dataset. The most important part of this task is likely the creation of a training instrument: a set of questions to ask humans to code about the objects to be analyzed. But here we assume the analyst has an instrument at hand and focus on the question of how to apply the instrument.[23] To do so, the analyst must: a) choose a unit of analysis for coding; b) choose coders; and c) decide how many coders to have code each document.[24]

To understand the significance of these decisions, recall that the purpose of the training data is to train a classifier. We estimate a model of sentiment (Y) as labeled by humans as a function of the text of the objects, the features of which compose the independent variables. Our goal is to develop a model that best predicts the outcome out of sample. We know that to get the best possible estimates of the parameters of the model we must be concerned with measurement error about Y in our sample, the size of our sample, and the variance about our independent variables. Since, as we see below, measurement error about Y will be a function of the quality of coders and the number of coders we use per object, it is impossible to consider quality of coders, number of coders and size of the training set independently. Given the likely existence of some budget constraint we will need to make a choice between more coders per object and more objects coded. In addition, the unit of analysis (e.g., sentences or articles) selected for human coding will affect the amount of information contained in the training dataset, and thus affect the precision of our estimates.

In what follows we present the theoretical trade-offs associated with the different choices an analyst might make when confronted with the decisions listed above. We present empirical evidence and offer guidelines for the analyst seeking to build an effective training dataset. Our goal in the running example is to develop the best measure of tone of coverage of the U.S. national economy in the *New York Times* over the period 1947 to 2014, where *best* refers to the measure that best predicts the tone as perceived by

[23] The creation of a training instrument is covered voluminously in other contexts and is beyond the scope of our work here. A combination of methods from survey research and content analysis are relevant. Seminal works include (Groves et al. 2009), (Sudman, Bradburn & Schwartz 1995), and (Bradburn, Sudman & Wansink 2004), and Krippendorff:2018.

[24] We discuss a method of choosing coders based on comparison of their performance and cost in section 6 of the Appendix. And we note that this is not an exhaustive list of the necessary steps to create a training dataset.

human readers of the text. Throughout, unless otherwise noted, we use a binary classifier trained from coding produced using a 9-point ordinal scale (where 1 is very negative and 9 is very positive) collapsed such that 1-4=0, 6-9=1. If a coder used the midpoint (5), we treated the coding as the equivalent of irrelevant and did not use the item in our training dataset. 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 three documents and no more than 80% of all documents (stopwords are included).[25] Analyses in each section draw on a number of different training datasets where the sample size, unit of analysis coded, the type and number of coders, and number of objects coded vary in accord with the comparisons of interest. Each dataset is named for the sample of objects coded (identified by a number from 1 to 5 for our 5 samples), the unit of analysis coded (S for sentences or A for article-segments), and the coders used (U for undergraduates and C for CrowdFlower workers). For example, Dataset 5AC denotes sample number five, and is based on article-segment-level coding by crowd coders. (See Appendix Table 1 in Appendix for details.) We never asked coders to read the entire article. Rather, based on the relatively standard reporting used by major newspapers we assume that the tone of the first five sentences of the article will represent the tone of the entire article, and we thus select approximately the first five sentences of an article for our coders to code.[26] We refer to these as article-segments, though they are always the beginning of the article.

For the purpose of assessing out-of-sample accuracy we have two “ground truth” datasets. The first (which we refer to as CF Truth) was produced by having each of 4400 article-segments randomly selected from the corpus coded by ten CrowdFlower work-

[25] We compared the performance of a number of classifiers with regard to accuracy and precision in both out-of-sample and cross-validated samples before selecting logistic regression with an L2 penalty. See Figures 1 and 2 in Section 4 of the Appendix for details.

[26] We attempt to select the first five sentences for each article. In cases where we were working with original PDFs transferred to text via OCR, errors in the OCR translation would sometimes result in initial segments of more than five sentences.

ers. We then utilized the set of 442 articles that were coded as relevant by at least seven of the ten coders. We define CF Truth as the average tone coded for each of these articles. If the average coding was neutral (5), the article was omitted from CF Truth. The second "ground truth" dataset (which we refer to as UG Truth) is based on Dataset 3SU (Appendix Table 1) in which between 2 and 14 undergraduate students coded 4,195 sentences individually using a 5 category coding scheme (negative, mixed, neutral, not sure, positive) from articles selected at random from the corpus.[27] We defined each sentence as positive or negative based on a majority rule among the codings (if there was a tie, then the sentence is coded as neutral or mixed). The tone of the article was defined by aggregating the individual sentences coded in the article, again following a majority rule so that an article is coded as positive in UG Truth if a majority of the first five sentences classified as either positive or negative are classified as positive.

3.1 Selecting a Unit of Analysis: Segments vs. Sentences

Should a supervised machine learning classifier be trained using coding that matches the unit of interest to be classified (e.g., an entire news article), or a smaller unit within it (e.g., a sentence)? Arguably the dictum "code the unit of analysis to be classified" should be the default position: if we wish to code articles for tone, we should train the classifier based on article-level human coding. Indeed, we have no reason to expect that people reading an article come away assessing the tone of the entire article as a simple sum of its component sentences.

However, our goal in developing a training dataset is to obtain estimates of the weights to assign to each feature in the text in order to predict the tone of an article. There are at least two reasons to think that sentence-level coding may be a better way to achieve this goal. First, if articles contain sentences that are not relevant to the tone of

[27] Variation in number of coders was a function of how many undergraduates completed tasks. The 5 category coding scheme used here was employed before switching to the 1-9 scheme later in the project.

the article, these would add noise to article-level coding, since features in these irrelevant sentences would contribute to an article-level classifier's training. But using sentence-level coding, these irrelevant sentences can be excluded. Second, if individual sentences contain features with a single valence (i.e., either all positive or all negative), but articles contain both positive and negative sentences, then information will be lost if the coder must choose a single label for the entire article. Of course, if articles consist of uniformly toned sentences, then any benefit of coding at the sentence level is likely lost. Empirically it is an open question whether we would be better off coding sentences or articles.

Here, we do not compare sentence-level coding to article-level coding directly, but rather compare sentence-level coding to "segment"-level coding, using the first five or so sentences in an article as a segment. Although a segment as we define it is not nearly as long as an article, it retains the key distinction that underlies our comparison of interest, namely that it contains multiple sentences. And, of course, compiling a training dataset of segments was much more cost effective than coding full articles.

Below we discuss the distribution of relevant and irrelevant sentences within relevant and irrelevant segments in our data, and then discuss the distribution of positive and negative sentences within positive and negative segments.[28] Then we compare the out-of-sample predictive accuracy of a classifier based on segment-level coding to a classifier based on sentence-level coding. We evaluate the effect of unit of analysis using two training datasets. In the first (Dataset 1SC in Appendix Table 1), three CrowdFlower coders coded 2,000 segments randomly selected from the corpus. In the second (Dataset 1AC) three CrowdFlower coders coded each of the sentences in these same segments individually.[29]

We first compute the average number of sentences coded as relevant and irrelevant

[28] See Section 5 of the Appendix for discussion of other considerations.

[29] Coding was conducted using our 9-point scale. Sentences were randomized, so individual coders were not coding sentences grouped by segment.

in cases where an article was unanimously coded as relevant by all three coders. We find that on average slightly more sentences are coded as irrelevant (2.64) as opposed to relevant (2.33) (see Appendix Table 6). This finding raises concerns about using segments as the unit of analysis, since a segment-level classifier would learn from features in the irrelevant sentences, while a sentence-level classifier could ignore them.

Next, in Table 3 we present the distribution of the average count of positive and negative sentences for the subset of 1,790 articles coded as relevant by at least one coder. We find that among the set of segments all coders agreed were positive, an average of just under one sentence (0.91) was coded positive by all coders, while fewer than a third as many sentences were on average coded as having negative tone (0.27). Negative segments tended to contain one (1.00) negative sentence and essentially no positive sentences (0.08). The homogeneity of sentences within negative segments suggests coding at the segment level might do very well. The results are more mixed in positive segments.[30] If we have equal numbers of positive and negative segments, then approximately one in five negative sentences will be contained in a positive segment. That could create some error when coded at the segment level.[31]

Finally, in order to assess the performance of classifiers trained on each unit of analysis, we produce two classifiers: one by coding tone at the sentence level and one by coding tone at the segment level.[32] We compare out-of-sample accuracy of segment classification based on each of the classifiers using the CF Truth dataset (where accuracy is measured at the segment level). The out-of-sample accuracy scores of the data coded at

[30] This asymmetry is intriguing, and worthy of additional future study.

[31] If we drop the unanimity threshold to coding based on majority-rule, we see more positive sentences in negative segments, and more negative sentences in positive segments. See Tables 3, 4, and 5 in Section 5 of the Appendix to see how this distribution varies with different assignment rules based on multiple codings.

[32] We treat a sentence or segment as relevant if at least one coder codes it as relevant, and we only use objects (sentences or segments) coded as relevant. In the segment-level dataset, at least one coder coded 1,790 of the segments as relevant, and an average of 2.26 coders coded each segment as relevant. In the sentence-level dataset, at least one coder coded 8,642 sentences as relevant, with an average of 2.16 coders marking each sentence as relevant.

Table 3: Article by Sentence Tone

	Average # of Positive Sentences	Average # of Negative Sentences
Positive Articles	0.91	0.27
Negative Articles	0.08	1.00

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

the sentence and segment levels are 0.700 and 0.693, respectively. The choice of unit of analysis to code has, in this case, surprisingly little consequence, suggesting there is little to be gained by going through the additional expense and complexity of coding segments at the sentence level. Because more segments can be coded for the same level of resources while also reducing the processing burden associated with breaking larger units into sentences, we are better off coding segments in our example. While some other dataset might lead to a different outcome, we think analysts could proceed with segment-level coding.

Coding Segments vs. Coding Sentences

Advantages:
Sentences: Irrelevant sentences can be ignored (whereas irrelevant sentences within segments will add noise).
Segments: Easier and cheaper to code than sentences, and in many cases this larger unit is what we wish to classify.

Findings: In our test, the tone of a segment tends to be consistent with the predominant tone of its sentences. But regardless of a segment's tone, more than half of the sentences tend to be irrelevant, suggesting that segment-level coding may be noisier. However, classifiers trained on sentences and on segments performed nearly identically in accurately predicting segment-level tone, suggesting that the easier and cheaper segment-level coding is preferable.

Advice: Code by segment, unless there is reason to suspect wide variance in tone across sentences, and/or a high proportion of irrelevant sentences, within segments.

3.2 Allocating Total Codings: More Documents vs. More Coders

Having decided the unit to be coded, the analyst faces another decision based on the reality of having some budget constraint: should each coder label a unique set of documents and thus have one coding per document, or should multiple coders code the same set of documents to produce multiple codings per document, but on a smaller set of documents? To provide an example of the problem, assume four coders of equal quality. Further assume an available budget of $100 and that each document coded by each coder cost ten cents such that the analyst can afford 1,000 total codings. If the analyst uses each coder equally, e.g. each will code 250 documents, the relevant question is whether each coder should code 250 unique documents, all coders should code the same 250 documents, the coders should be distributed such that two coders code one set of 500 unique documents and the other two coders code a different set of 500 unique documents, and so on.[33]

The answer is readily apparent if the problem is framed in terms of levels of observations and clustering. If we have multiple coders coding the same document, then we have only observed one instance of the relationship between the features of the document and the *true* outcome, though we have multiple *measures* of it. Thus estimates of the classifier weights will be less precise, i.e., $\hat{\beta}$ will be further from the truth, and our estimates of $\hat{Y}$, the sentiment of the text, will be less accurate than if each coder labelled a unique set of documents and we expanded the sample size of the set where we observe the relationship between the features of documents and the true outcome. In other words, coding additional documents provides more information than does having an additional coder, coder i, code a document already coded by coder j. Consider the limiting case where all coders code with no error. Having a second coder code a document provides *zero* information and cannot improve the estimates of the relationship between the features of

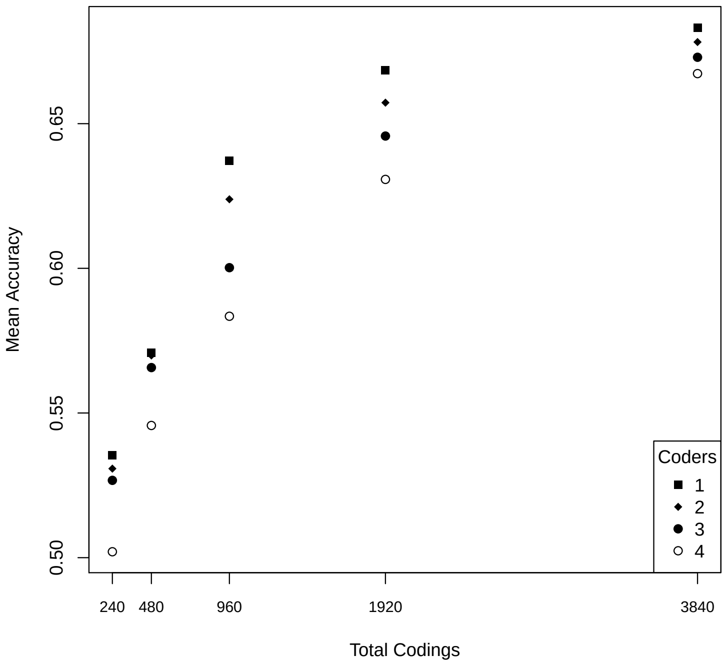
[33] One can create more complex schemes that would allocate a document to two coders, and only go to additional coders if there is disagreement. Here we only consider cases where the decision is made ex-ante.

the document and the outcome. However, coding an additional document provides one new datapoint, increasing our sample size and thus our statistical power. Intuitively, the benefit of more documents over more coders would increase as coder accuracy increases.

We performed several simulations to examine the impact of coding each document with multiple coders versus more unique documents with fewer coders per document. The goal is to achieve the greatest out-of-sample accuracy with a classifier trained on a given number of *Total Codings, TC*, where we vary the number of *unique documents, n*, and the number of *coders, j,* per document. We present results from a simulation showing the impact of varying the number of coders and number of documents coded for a binary classification task. To mimic our actual coding tasks, we generate 20,000 documents that have a true value between 0 and 1 based an underlying linear model using 50 independent variables, converted to a probability with a logit link function. We then simulate unbiased coders with variance of 0.81 to produce a continuous coding of a subset of documents.[34] Finally, we convert each continuous coding to a binary (0/1) classification. Using these codings we estimate an L2 logit.

[34] See section 6 in the Appendix for a discussion of describing coder quality by the variance of individual coders.

Figure 1: **Accuracy with Constant Number of Total-Codings**



Note: Results are based on simulations described in the text. Plotted points are jittered based on the difference from mean to clearly indicate ordering.

Figure 1 shows accuracy based on a given number of total codings TC achieved with different combinations of number-of-coders, j, and number-of-unique-documents, n, where $n \in$ {240, 480, 960, 1920, 3840} and j ranges from 1 to 4. For example, the first vertical set of codings shows mean accuracy rates achieved with one coder coding 240 unique objects, 2 coders coding 120 unique objects, 3 coders coding 80 unique objects, and 4 coders coding 60 unique objects. The results demonstrate that for any given number of total codings, TC, predictive accuracy is always higher the fewer coders: $PCP_{TC|j} > PCP_{TC|(j+k)} \quad \forall \, k > 0$.

The simulation demonstrates that the analyst seeking to optimize predictive accuracy for any fixed number of total codings should maximize the number of unique documents coded. While increasing the number of coders for each document can improve the accuracy of the classifier (see Section 7 in the Appendix), the informational gains from increasing the number of documents coded are greater than that from increasing the number of codings of a given document. Thus, given a budget constraint that limits the number of total codings, the optimal strategy is to have each coder code a unique set of documents. This does not obviate the need to have multiple coders code at least a subset of documents, namely to determine coder quality and select the best set of coders to use for the task at hand. But once the better coders are identified, the optimal strategy is to proceed with one coder per document.

Allocating Total Codings: More Documents vs. More Coders

Advantages*:
More Unique Documents (thus having fewer coders per document): More information in the training dataset, and thus higher precision of the classifier.
More Coders per Document (thus coding fewer unique documents): Allows identification of better/worse coders.

** These tradeoffs assume a fixed amount of resources and a fixed number of codings, remembering that increasing coders or documents will always improve predictive accuracy.*

Findings: Simulations show that, given a fixed number of codings, accuracy is always higher by maximizing the unique number of documents coded.

Advice: For any fixed number of total codings, maximize the number of unique documents coded.

4 Selecting a Classification Method: Supervised Machine Learning vs. Dictionaries

Dictionary methods and supervised machine learning methods constitute the two primary approaches for coding the tone of large amounts of text. Here we describe each classification method, discuss the advantages and disadvantages of each, and assess the ability of a number of dictionaries and SML classifiers (1) to correctly classify documents labeled by humans and (2) to distinguish between more and less positive documents.[35]

A dictionary is a user-identified set of features or terms relevant to the coding task where each feature is assigned a weight that reflects the feature's user-specified contribution to the measure to be produced, usually +1 for positive and -1 for negative features. The analyst then applies some decision rule, such as summing over all the weighted feature values, to create a score for the document. By construction, dictionaries code documents on an ordinal scale. In other words, they sort documents as to which are more or less positive or negative relative to one other. If an analyst wants to know which articles are positive or negative, they need to identify a cutpoint. We may assume that an article with more positive terms than negative terms is positive, but we do not know ex-ante if human readers would agree. If one is interested in relative tone, e.g., if one wanted to compare the tone of documents over time or the relationship between tone and other variables over time, the uncertainty about the zero-point is not an issue.

The analyst selecting supervised machine learning (SML) methods follows three broad steps. First, a sample of the corpus (the training dataset) is coded (classified) by humans for tone, or whatever aspect of the text is being measured (the text is labeled). Then a classification method (machine learning algorithm) is selected and the classifier is trained to predict the label assigned by the coders within the training dataset.[36] In this way the classifier 'learns' the relevant features of the dataset and how these features are related to the labels. Multiple classification methods are generally applied to the data and tested for minimum levels of accuracy using cross-validation to determine the best classifier. Finally, the chosen classifier is applied to the entire corpus of text being analyzed to predict the sentiment of all unclassified articles (those not labelled by humans).

[35] Note that, depending on the task at hand, analysts may choose to use SML for one task and dictionaries for another (see, for example, Stecula and Merkley (2019)).

[36] The terminology 'training a classifier' is unique to machine learning, but easily translates to traditional econometrics as 'choose the model specification and estimate model parameters'.

Dictionary and SML methods allow analysts to code vast amounts of text that would not be possible with human coding and each presents unique advantages but also challenges. One advantage to a dictionary approach is that many dictionaries have already been created for a variety of tasks, including measuring the tone of text. If an established dictionary is a good fit for the task at hand, then it is relatively straightforward to apply it. However, if an appropriate dictionary does not already exist, the analyst must create one. Because creating a dictionary requires identifying features and assigning weights to them, it is a difficult and time consuming task. Fortunately, humans have been 'trained' on a lifetime of interactions with language and thus can bring a tremendous amount of prior information to the table to assign weights to features. Of course, this prior information meets many practical limitations. Most dictionaries will code unigrams, since if the dictionary is expanded to include bigrams or trigrams the number of potential features increases very quickly and adequate feature selection becomes untenable. And all dictionaries necessarily consider a limited and subjective set of features, meaning not all features in the corpus and relevant to the analysis will be in the dictionary. It is important, then, that analysts taking a dictionary approach carefully vet their selection of terms. For example, Muddiman and Stroud (2017) construct dictionaries by asking human coders to identify words for inclusion in each dictionary, and then calculated the inter-coder reliability of the coders' suggestions. Further, in assigning weights to each feature, analysts must make the assumption that they know the importance of each feature in the dictionary and that all text not included in the dictionary has no bearing on the tone of the text.[37] Thus, even with rigorous validation, dictionaries necessarily limit the amount of information that can be learned from the text.

In contrast, when using SML methods the relevant features of the text and their weights are estimated from the data.[38] The feature space is thus likely to be both larger

[37] Some dictionaries, e.g., SentiStrength, allow users to optimize weights using a training set, which is essentially using SML with a restricted feature set.

[38] The analyst is not prohibited from bringing prior information to bear by, for example, including pre-

and more comprehensive than that used in a dictionary. Further, SML can more readily accommodate the use of n-grams or co-occurrences as features and thus partially incorporate the context in which words appear. Finally, since supervised machine learning methods are trained on data where humans have labelled an article as ‘positive’ or ’negative’, SML methods estimate a true ‘zero’ point and can classify individual documents as positive or negative. The end result is that much more information drives the subsequent classification of text.

But SML presents its own challenges. Most notably it requires the production of a large training dataset coded by humans and built from a random set of texts in which the features in the population of texts are well represented. Creating the training dataset itself requires the analyst to decide a unit of analysis to code, the number of coders to use per object, and the number of objects to be coded. These decisions, as we show above, can affect the measure of tone produced. In addition, it is not clear how generalizable any training dataset is. For example, it may not be true that a classifier trained on data from the *New York Times* is optimal for classifying text from *USA Today* or that a classifier trained on data from one decade will optimally classify articles from a different decade.

Dictionary methods allow the analyst to bypass these tasks and their accompanying challenges entirely. At the same time, however, the production of a human coded training dataset for use with SML allows the analyst to evaluate the performance of the classifier with measures of accuracy and precision using cross-validation. Analysts using dictionary methods typically have no (readily available) human coded documents with which to evaluate performance of the dictionary classification. Even when using dictionaries that have been tested by their designers, there is no guarantee that the test of the dictionary on one corpus for one task or within one general domain (such as newspaper articles) validates the dictionary’s use on a different corpus for a different task or domain (e.g. tweets). In fact the evaluation of the accuracy of dictionaries is difficult precisely

specified combinations of words as features whose weights are estimated from the data.

because of the issue discussed earlier, that they have no natural cut point to distinguish between positive documents and negative documents. Thus the only way to evaluate performance of a dictionary is to have humans code a sample of the corpus and examine whether the dictionary assigns higher scores to positive documents and lower scores to negative documents as evaluated by human coders. This is what Young and Soroka (2012) do in the evaluation of Lexicoder: they bin documents based on scores assigned by human coders and report the average Lexicoder score for documents in each bin. By showing that the Lexicoder score for each bin is correlated with the human score, they validate the performance of Lexicoder.[39] Analysts appropriating dictionaries 'off-the-shelf' could perform a similar exercise for their applications, but at that point the benefits of using a dictionary begin to deteriorate. In any case, analysts using dictionaries should take care both in validating the inclusion of terms to begin with (e.g., Muddiman and Stroud (2017) *and* validating that text containing those terms has the intended sentiment (e.g., Young and Soroka (2012)).

Given the advantages and disadvantages of the two methods, how should the analyst think about which is likely to perform better? Below we compare their performance empirically in our example. However, before moving to the empirics, we can perform a thought experiment. If we assume both the dictionary and the training dataset are of high quality, then we already know that if we consider SML classifiers that utilize only words as features, it is mathematically impossible for dictionaries to do as well as a SML model trained on a large enough data set if we are testing for accuracy *within* sample. The dictionary comes with a hard-wired set of parameter values for the importance of a pre-determined set of features. The SML model will estimate parameter values optimized to minimize error of the classifier on the training dataset. Thus, SML will necessarily outperform the dictionary on that sample. So the relevant question is, which does better *out*

[39] As another example, Thelwall et al. (2010) compare human coding of short texts in MySpace with each positive and negative SentiStrength scores to validate their dictionary.

of sample? Here, too, since the SML model is trained on a sample of the data, it is guaranteed to do better than a dictionary *as long as it is trained on a large enough random sample*. As the sample converges to the population—or as the training dataset contains an ever increasing proportion of words encountered—SML has to do better than a dictionary, as the estimated parameter values will converge to the true parameter values.

While a dictionary cannot compete with a classifier trained on a representative and large enough training dataset, in any given classification task dictionaries may out-perform SML if the training dataset is not representative and/or not large enough. Dictionaries bring rich prior information to the classification task: humans may produce a topic-specific dictionary that would require a large training dataset to out-perform it. Similarly, a poor training dataset may not contain enough (or good enough) information to out-perform a given dictionary. So if the analyst cannot create a sufficiently high-quality training dataset, a dictionary may prove superior at classification. Below we compare the performance of a number of dictionaries with SML classifiers in the context of coding sentiment about the economy in the *New York Times*, and we examine the role of the size of the training dataset set in this comparison in order to assess the utility of both dictionaries and SML.

4.1 Comparing Classifiers

The first step in comparing the two approaches is to identify the dictionaries and classifiers we wish to compare. We consider two widely used sentiment dictionaries. The first of these, SentiStrength, is a general sentiment dictionary optimized for short texts (Thelwall et al. 2010). The second dictionary, Lexicoder, is a sentiment dictionary designed specifically to be applied to political text (Young & Soroka 2012). In brief, application of SentiStrength produces a positive and negative score for each document based on the word score associated with the strongest positive word (between 0 and 4) and the strongest negative word (between 0 and -4) in the document that are also contained in the dictionary. The authors did not choose to generate a net tone score for each document. We do so by summing these two positive and negative sentiment scores, taken from the strongest positive and negative word in each document, such that document scores range from -4 to +4. In contrast, Lexicoder assigns every n-gram in a given text a binary indicator if that n-gram is in its dictionary, coding for whether the n-gram is positive or negative. Sentiment scores for documents are then calculated by taking the number of positive terms minus the number of negative terms divided by the total number of terms in the document. We also consider the relatively simple 'dictionary' proposed by Hopkins et al. (2017), which contains just 21 economic terms.[40] Hopkins et al. did not generate article-level scores, but using their dictionary of terms they calculate the fraction of articles per month mentioning each of the terms. The fractions are summed, with positive and negative words having opposite signs, to calculate net tone in a given time interval. We extend their logic to predict article-level scores by summing the number of unique positive stems and subtracting the number of unique negative stems in an article to produce a measure of sentiment.

We consider two SML classifiers, each trained on a dataset generated from 4,400 unique articles (Dataset 5ac in Appendix Table 1) in the *New York Times* randomly sampled from the years 1947 to 2014.[41]

Between three and ten CrowdFlower workers coded each article for relevance. At

[40] Based on an iterative procedure designed to maximize convergent validity, Hopkins and co-authors chose to use 15 negative terms: 'bad', 'bear', 'debt', 'drop', 'fall', 'fear', 'jobless', 'layoff', 'loss', 'plung', 'problem', 'recess', 'slow', 'slump', 'unemploy', and 6 positive terms: 'bull', 'grow', 'growth', 'inflat', 'invest', and 'profit'.

[41] Selecting the optimal classifier to compare to the dictionaries requires a number of decisions that are beyond the scope of this paper (but see Raschka (2015), James (2013), Hastie (2009), Caruana (2006)), including how to preprocess the text—whether to stem the text (truncate words to their base), how to select and handle stopwords (commonly used words that do not contain relevant information), and the nature and number of features (n-grams) of the text to include. Denny and Spirling (2018) show how the choice of preprocessing methods can have profound consequences, and we examine the effect of some of these decisions on accuracy and precision in Section 4 of the Appendix. But in the body of the text here we focus on other issues, assuming pre-processing decisions have been made.

least one worker coded 4,070 articles as relevant with an average of 3.53 coders coding each relevant article for tone using the 9-point coding instrument. The optimal classifier was selected from a set of single-level classifiers including logistic regression (with L2 penalty), Lasso, ElasticNet, SVM, Random Forest, and AdaBoost.[42] Based on accuracy and precision evaluated using UG Truth and CF Truth, we selected the regularized logistic regression with L2 penalty with up to 75,000 n-grams appearing in at least 3 documents and no more than 80% of all documents, including stopwords, and stemming.

We report the n-grams that are most predictive of positive and negative tone in Table 4. 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.

We can now compare the performance of each dictionary and classifier. We begin by assessing the accuracy and precision of each classification method with respect to the tone of articles in both the UG Truth and CF Truth datasets, comparing the percentage of articles for which each approach correctly predicts the direction of tone and for which each correctly predicts individual articles as positive that were coded as positive by humans. We then consider the role of the size of the training dataset used to train the SML classifier, as well as the threshold selected for classification, in these comparisons. Then we

[42] One could attempt to simultaneously model relevance and tone, or to simultaneously model topics and then assign tone within topics—allowing the impact of words to vary by topic. Those are considerations for future work.

Table 4: **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.

separately assess accuracy for the baseline SML classifier and Lexicoder for articles humans have coded as particularly negative or positive and those about which our coders are more ambivalent.[43]

4.1.1 Accuracy and Precision

We begin by assessing accuracy (the percent correctly classified) and precision (the percent of positive articles classified as positive) of each of our classifiers on our CF Truth dataset.[44] Figure 2 presents our results. The left panel presents the accuracy of the classifiers while the right panel presents their relative precision. We include a dotted line

[43] See Section 9 in the Appendix for a comparison of the relationship between monthly measures of tone produced by each classification method and standard measures of economic performance. These comparisons demonstrate the convergent validity of the measures produced by each classifier.

[44] All articles for which the SML classifiers generated a probability of being positive greater than 0.5 were coded as positive. For each of the dictionaries we coded an article as positive if the sentiment score generated by the dictionary was greater than zero. This assumes an article with more positive (weighted) terms than negative (weighted) terms is positive. This rule is somewhat arbitrary and different decision rules will change the accuracy (and precision) of the classifier. The ROC curve presented below illustrates the impact of changing the cutpoint.

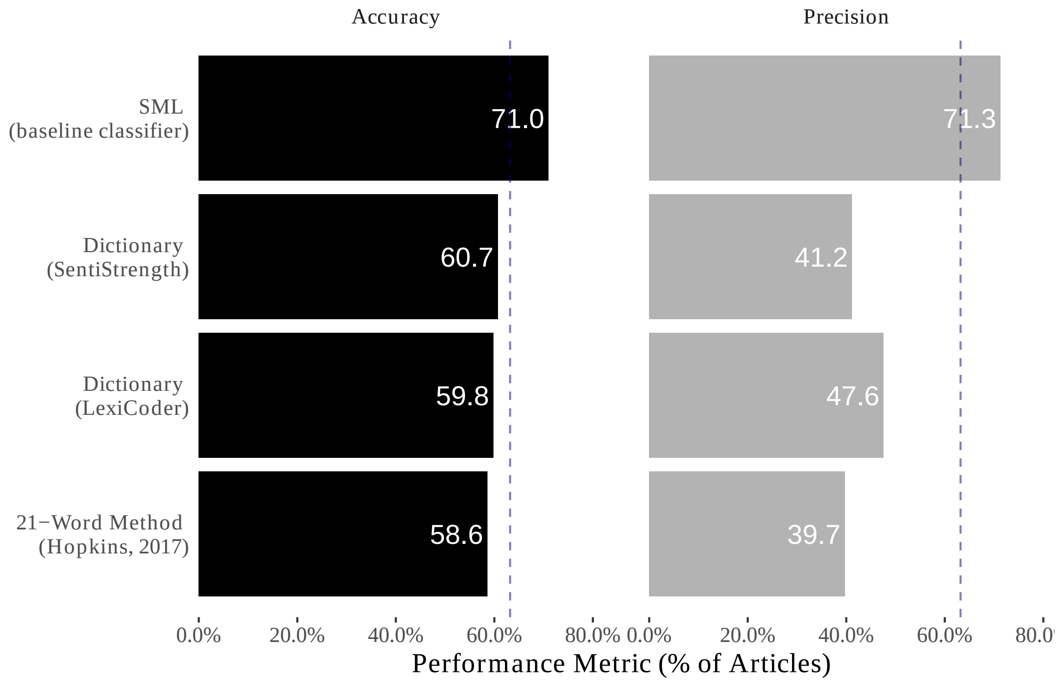
in each panel of the figure to represent the percentage of articles in the modal category. Any classifier can achieve this level of accuracy simply by *always* assigning each document to the modal category. Not only do machine learning approaches out-perform dictionary methods in terms of accuracy, *only* machine learning classifiers out-perform the naive guess of the modal category. More specifically, we find that our baseline SML classifier correctly predicts coding by crowd workers in 71.0% of the articles they coded. In comparison, SentiStrength correctly predicts 60.7%, Lexicoder 59.8% and the Hopkins 21-Word Method 58.6% of the articles in CF Truth. The relative performance of the SML classifiers is even more pronounced with respect to precision, which is the more difficult task here as positive articles are the rare category. Our baseline SML model correctly predicts positive articles 71.3% of the time while SentiStrength does so 41.2% of the time and Lexicoder and Hopkins 21-Word Method do so 47.6% and 39.7% of the time, respectively. In sum, each of the dictionaries is 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.[45]

What is the role of the size of the training dataset in explaining the better accuracy and precision rates of the machine learning classifiers? To answer this question, we drew 10 random samples of 250 articles each from the full CF Truth training dataset. Using the same method as above, 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.[46] We repeated this process for sample sizes of 250 to 8,750 by increments of 250. The accuracy results are presented in Figure 3, with shaded areas indicating the 95% confidence interval. The x-axis gives the size of the training data set and the y-axis reports the average

[45] Similar accuracy and precision are obtained with respect to the UG Truth dataset. Results are presented in the Appendix.

[46] Results for recall and precision rates by training dataset size may be found in Section 10 of the Appendix. Briefly, we find that recall—the fraction of positive articles correctly coded as positive by our classifier—behaves similarly to accuracy. However, precision—the fraction of articles we predict as positive that coders identified as being positive—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 2: **Performance of Machine Learning and Dictionary Methods—Accuracy and Precision**

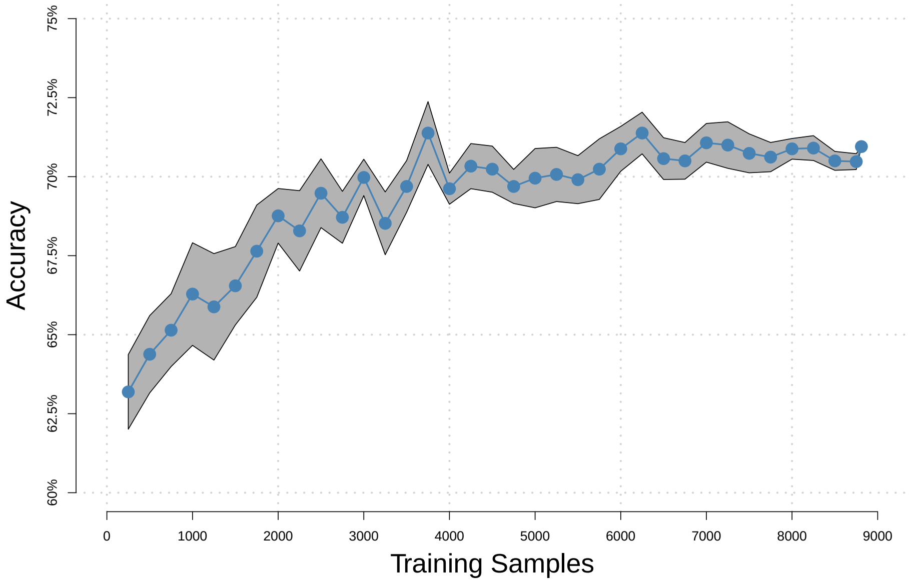


Note: Accuracy (percentage of articles correctly classified) and precision (percentage of positive articles predicted to be positive) are reported for the ground truth dataset coded by 10 CrowdFlower coders. The dashed vertical lines indicate the baseline level of accuracy if the modal category is always 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).

accuracy in CF Truth for the given sample size. The final point represents the full training dataset, and as such there is only one accuracy rate (and thus no confidence interval).

What do we learn from this exercise? Using the smallest training dataset (250), the accuracy of our baseline classifier equals the percent of articles in the modal category (about 63%). Further, accuracy improves quite quickly as the size of the training dataset increases. With 2,000 observations, the SML classifier is quite accurate, and there appears to be very little return for a training dataset with more than 3,000 articles. While it is clear that in this case 250 articles is simply not a large enough training dataset to develop an accurate SML classifier, even using this small training dataset the SML classifier has greater accuracy with respect to CF Truth than that obtained by any of the dictionaries.[47]

Figure 3: **Accuracy 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 5AC, Appendix 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 correctly classified.

An alternative way to compare the two classifiers is to use a receiver operator characteristic, or ROC, curve. An ROC curve allows us to assess the ability of each classifier to correctly predict whether the tone of an article is positive in CF Truth for any given classification threshold. In other words, it provides a visual description of a classifier's ability to separate negative from positive articles across all possible classification rules. Figure 4

[47] The results of this exercise do not suggest that a training dataset of 250 will consistently produce accuracy rates equal to the percent in the modal category, nor that 2,000 or even 3,000 observations is adequate to the task in any given application. The size training dataset required will depend both on the quality of the training data, likely a function of the quality of the coders and the difficulty of the coding task, as well as the ability of the measured features to predict the outcome.

presents the ROC curve for our baseline SML classifier and for the Lexicoder dictionary.[48] The x-axis in the ROC curve gives the false positive rate—the proportion of all negatively toned articles in CF Truth that were classified as positively toned—and the y-axis gives the true positive rate—the proportion of all positively toned articles in CF Truth that were classified as positive. Each point on the ROC curve represents the misclassification rate for a given classification threshold. Two things are of note. First, for almost any classification threshold we might choose, the SML classifier gives a higher true positive rate than Lexicoder. Only in the very extreme cases in which articles are classified as positive only if the predicted probability generated by the classifier is very close to 1.0 (in the top right corner of the figure) does Lexicoder misclassify articles slightly less often. Second, the larger the area under the ROC curve (AUC), the better the classifier's performance. In this case the AUC of the SML classifier (0.752) is significantly greater (p=0.00) than for Lexicoder (0.603). This finding confirms that the SML classifier has a greater ability to distinguish between more positive versus less positive articles.

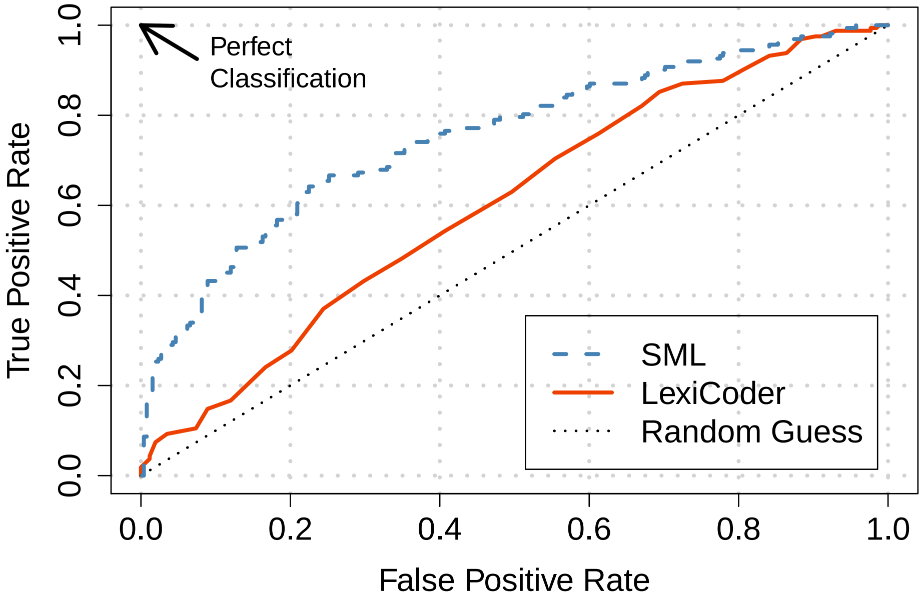
4.1.2 Ability to Discriminate

One potential shortcoming of focusing on predictive accuracy may be that, even if SML is better at separating negative from positive articles, perhaps dictionaries are better at capturing the gradient of potential values of sentiment, from very negative to very positive. If this were the case, then dictionaries could do well when comparing the change in sentiment across articles or between groups of articles. In fact, this is what we are often trying to do when we examine changes in tone from month to month.

To examine how well each method gauges *relative* tone, we conduct an additional validation exercise similar to that performed by Young and Soroka (2012) to assess the performance of Lexicoder relative to our baseline SML classifier. Instead of reporting

[48] Lexicoder scores were standardized to range between zero and one for this comparison.

Figure 4: **Receiver Operator Characteristic Curve: Lexicoder vs. Supervised Machine Learning Classification**

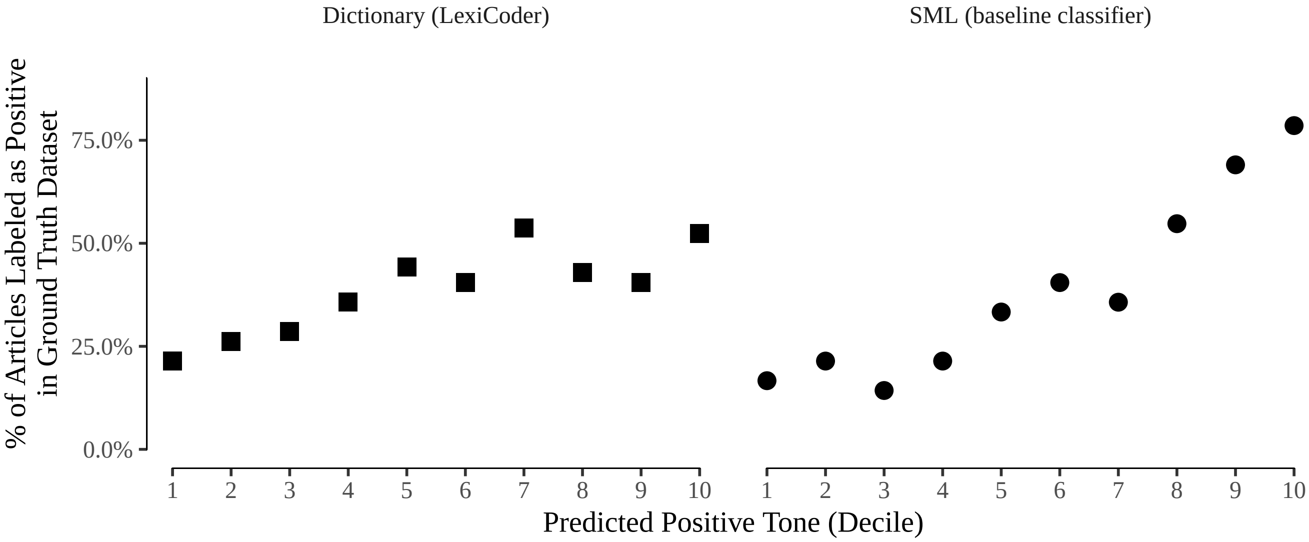


Note: The x-axis in the ROC curve gives the false positive rate—the proportion of all negatively toned articles in CF Truth that were classified as positively toned—and the y-axis gives the true positive rate—the proportion of all positively toned articles in CF Truth that were classified as positive. Each point on the ROC curve represents the misclassification rate for a given classification threshold. The corpus used in the analysis is based on the keyword search of *The New York Times* 1980-2011 (see the text for details).

accuracy at the article level, we split our CF Truth sample into sets of deciles according to (1) the sentiment score assigned by Lexicoder and (2) the predicted probability according to our baseline SML classifier. We then measure the proportion of articles that crowd workers classified as positive within each decile. In other words, we look at the 10% of articles with the lowest sentiment score according to each method and count how many articles in CF Truth are positive within this bucket; and then repeat this step for all other deciles.

As Figure 5 shows, while in general articles in each successive bin according to the dictionary scores were more likely to have been labeled as positive in CF Truth, the differences are not as striking as with the binning according to SML. The groups of articles the dictionary places in the top five bins are largely indistinguishable in terms of the percent of articles labeled positive in CF Truth and only half of the articles with the highest dictionary scores were coded positive in CF Truth. In contrast, when we look at the SML classifier, we see a clearer ability to distinguish the tone of articles for most of the range, and over 75% of articles classified with a predicted probability of being positive in the top decile were labeled as such in CF Truth. In short, even when it comes to the *relative* ranking of articles, the dictionary does not perform as well as our baseline SML classifier and it is unable to accurately distinguish less positive from more positive articles over much of the range of dictionary scores.

Figure 5: **Classification Accuracy in CF Truth as a Function of Article Score (Lexicoder) and Predicted Probability an Article is Positive (SML Classification)**



Note: Dictionary scores and SML predicted probabilities are assigned to each article in the CF Truth dataset. Articles are then assigned to a decile based on this score. Each block or circle on the graph represents accuracy within each decile, which is determined based on coding from CF Truth. The corpus used in the analysis is based on the keyword search of *The New York Times* 1980-2011 (see the text for details.)

4.2 Selecting a Classification Method: Conclusions from the Evidence

Across the range of metrics considered here, supervised machine learning almost always out-performed the dictionaries. In analyses based on a full training dataset—produced with either CrowdFlower workers or undergraduates—SML was more accurate and had greater precision than any of the dictionaries. Moreover, when testing smaller samples of the CF training dataset, the SML classifier was more accurate and had greater precision even when trained on only 250 articles. Further, our binning analysis with Lexicoder showed that Lexicoder was not as clearly able to distinguish the *relative* tone of articles in CF Truth as was our baseline classifier; and the ROC curve we presented shows that the accuracy of our baseline classifier out-performed that of Lexicoder regardless of the threshold used for classification.

Our advice to analysts is to use supervised machine learning techniques to develop measures of tone rather than to rely on dictionaries. In addition to out-performing the dictionaries in our analysis, three features of SML classifiers lead us to this recommendation. First, with a large enough training dataset, SML must mathematically out-perform dictionaries. And, in our analysis, the size of the training dataset required was not very large.

Selecting a Classification Method: SML Classifiers vs. Dictionaries

Advantages:
Supervised Machine Learning (SML): More comprehensive set of features used to classify text; optimized for current research question; built-in validation mechanism vis a vis human coding; by construction, the analyst knows the performance of the classifier based on multiple measures of fit (i.e, how closely the labels generated correspond to human coding); natural zero-point for determining overall tone of a given text; mathematically, SML necessarily out-performs dictionary methods given a large enough training dataset.
Dictionaries: Can be off-the-shelf; no creation of a training dataset required; easy to apply to a given corpus; built by humans who can bring domain expertise to bear.

Findings: In our tests, SML out-performs dictionary methods in terms of accuracy and precision, and the ability to discriminate between more and less positive articles. A relatively small training dataset produced a high-quality classifier.

Advice: Use SML if resources allow for the building of a high-quality training dataset. If using dictionaries, choose a dictionary appropriate to the task at hand, and validate the utility of the dictionary by confirming that a sample of dictionary-generated scores of text in the corpus conform to human coding of the text for the measure of interest.

5 Recommendations for Analysts of Text

Producing a scalar measure of some quantity of interest from text has become easy. Yet the analysis of text for meaningful social science research remains hard. For those engaged in the enterprise, the first decision made is how to sample from the universe of documents in an effort to generate a corpus that represents the population of relevant documents, which is followed by a decision as to how to select a classification approach. If using supervised machine learning, one must make a number of decisions about generating the training dataset. What is the optimal unit of analysis to code, who will code the data, and how many coders should code each text-object? To date, the consequences of these decisions have not been well understood and, as we have demonstrated, are not always benign. Our goal here has been to lay out the costs and benefits of a number of different strategies, and in doing so to offer analysts a guide to use in selecting between competing options. We summarize our recommendations below.

Selecting the Corpus: Our most striking, and perhaps surprising, finding is that something as simple as how one chooses the corpus of text to analyze can have huge consequences for the measure we produce. Specifically, when attempting to identify newspaper articles about a given topic, it would seem that either keyword searches or searches based on pre-defined categories (e.g., those offered by LexisNexis) *should* provide similar results. However, comparing articles returned by a keyword search with those returned by searching on subject categories, we found remarkably little overlap. The keyword-generated corpus was substantially larger, and had a higher proportion of articles relevant to our topic of interest, suggesting it was the better corpus for our purpose. Perhaps more importantly, we found that analyses based on the two distinct sets of documents produced very different measures of the quantity of interest: sentiment about the economy.

One cannot, of course, generalize from this case: it may be that other keyword searches would not out-perform subject-category data. However, we think one lesson is clear here: such searches *can* yield remarkably different corpora, and the implications for reproducible science are profound. Two analysts examining the same problem could come to very different conclusions simply because they began with different corpora of text.

Furthermore, as social scientists, we place high value on the ability to reproduce the corpus and to define a strategy that is transportable across media sources, including those in other countries, and over time. A keyword search provides a well-defined method for choosing the corpora; the analyst can describe it precisely and other analysts can replicate or modify it as they see fit. Proprietary subject-category searches meet none of these criteria. Searches based on labels provided by a proprietary media platform are black-boxes that provide no transparency to the analyst, and there is no guarantee that they will remain consistent (or even available) over time.

We recommend the analyst use keyword searches, and begin by applying both a relatively narrow keyword search and a broader keyword search and coding a sample of each set of texts for relevance. If the broader search returns more objects and relevance does not go down, the broader search should be adopted as using the narrower search would exclude relevant articles and potentially bias the resulting measure. If the narrower search produces a larger proportion of relevant texts, then the analyst must consider the tradeoff between the smaller set that can be more cheaply coded but *may* cause bias by ignoring articles that are *non-random* with respect to the measure of interest, and the larger set that is more expensive to code and will include more noise. The strategy of using keyword searches with tests for relevance allows the analyst to optimize the search criteria in designing the population corpus *prior* to the stage of producing the final training dataset and estimating the parameters of the classifier.

Creating a Training Dataset: How should the analyst select a unit of analysis, and select the number of coders per text-object to be coded?

When deciding on the unit to be coded, we found that coding article segments was more efficient for our task than coding sentences. Further, segment-level coding has the advantage that the human coders are working closer to the level of object that is to be classified (here, the article), and it has the non-trivial advantage that it is cheaper and more easily implemented in practice. Thus, while it is possible that coding at the sentence level would produce a more precise classifier in other applications, our results suggest that coding at the segment level seems to be the best default.

After developing a coding instrument using best extant practices, and choosing a set of coders, the best course of action in terms of classifier accuracy is to maximize the number of unique objects coded, irrespective of the selected coder pool or the application of interest. Doing so produces more efficient estimates than having additional coders code an object, i.e., there is more information in additional unique codings than repeated coding of a given object. This is especially true if you have very high-quality coders, which is more likely the easier the coding task. Of course using multiple coders per object to identify higher quality coders remains a useful way to choose coders.

Selecting a Classification Method: Our recommendation is to use supervised machine learning. We are data analysts, and we can learn from data. Using SML does require the production of a training dataset, which is a nontrivial effort. But the math is clear: given a large enough training dataset, supervised machine learning has to out-perform a dictionary. And as we showed above, in our case the size of the training dataset required was not very large.

If an analyst is to use a dictionary, it is essential that they be careful to validate the results produced for their application. The dictionary may be systematically measuring objects incorrectly compared to how humans would interpret the object. We note that both SML and dictionaries are being used as substitutes for careful human reading of text. Neither method is producing estimates of what human readers would code that are free of measurement error even in the best of circumstances. If an analyst applies a dictionary carefully and concludes based on validation that the dictionary is producing a valid measure that is a proxy (i.e., an unbiased estimate but measured with error) of the variable of interest, should they be satisfied? As in most cases of data analysis, it depends on what will be done with the measure produced.

We know that if our goal is to produce an estimate of a quantity of interest that is an outcome variable, measurement error will not bias any estimates we produce, but will decrease our efficiency—meaning we need larger datasets. But if we are computing a measure that will be used as an explanatory variable, then measurement error can be more dangerous, biasing our estimates. Here, we can not be saved by getting more data. Thus the analyst must think carefully about the purpose of the purpose of the measure being generated. These cautions apply as much to an analyst producing measures with supervised machine learning using a carefully produced, high-quality training dataset as to an analyst using a dictionary. In neither case is there a guarantee that the measure produced will be accurate enough to allow for capturing unbiased estimates of some quantity of interest. Reporting accuracy and precision without considering the implications of the values for the next stage of the analysis can lead the analyst to have misplaced confidence in the utility of the measure they have produced.

These are not small problems. A dictionary or a supervised machine learning classifier may well produce a measure that is correlated with our quantity of interest. But measurement error has real problems, and can propagate in pernicious ways in quantitative analysis. Say we are measuring the tone of economic news, and want to use those estimates of tone to determine the impact of tone on economic perceptions. Simply put, if we measure tone with error, we will not necessarily be able to capture the impact of tone on economic perceptions. Having bad measures hurts. Thus, in summary, it may be worth the cost and effort it takes to produce a high-quality training dataset and estimate a supervised machine learning model optimized to reduce error for the task at hand.

A Parting Word. The opportunities afforded by vast electronic text archives and machine classification of text for the measurement of a number of concepts, including tone, are in a real sense unlimited. In a rush to take advantage of the opportunities, it is easy to overlook some important questions and to under-appreciate the consequences of some decisions. It is straightforward for an analyst to acquire a corpus of text, apply a coding scheme, and produce a measure of the concept of interest. But careful consideration must be paid to the questions covered above in producing a machine-generated measure from text. Moreover, no machine-generated measure of a latent characteristic of text should be trusted if it has not been verified by comparing it to human perception of the text. Our two most important pieces of advice are these: (1) use transparent and reproducible methods in selecting a corpus; (2) classify by machine, but verify by human.

References

Atkinson, Mary Layton, John Lovett & Frank R Baumgartner. 2014. "Measuring the media agenda." *Political Communication* 31(2):355–380.

Bai, Jing, Dawei Song, Peter Bruza, Jian-Yun Nie & Guihong Cao. 2005. Query expansion using term relationships in language models for information retrieval. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM pp. 688–695.

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.

Blood, Deborah J & Peter CB Phillips. 1997. "Economic headline news on the agenda: New approaches to understanding causes and effects." *Communication and democracy: Exploring the intellectual frontiers in agenda-setting theory* pp. 97–113.

Bradburn, Norman M, Seymour Sudman & Brian Wansink. 2004. *Asking Questions: The Definitive Guide to Questionnaire Design*. John Wiley and Sons.

Caruana, Rich & Alexandru Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*. ACM pp. 161–168.

Condorcet, Marie J et al. 1972. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Vol. 252 American Mathematical Soc.

De Boef, Suzanna & Paul M Kellstedt. 2004. "The political (and economic) origins of consumer confidence." *American Journal of Political Science* 48(4):633–649.

Denny, Matthew James & Arthur Spirling. 2018. "Assessing the Consequences of Text Preprocessing Decisions." *Political Analysis* 26:168–189.

Doms, Mark E & Norman J Morin. 2004. "Consumer sentiment, the economy, and the news media." *FRB of San Francisco Working Paper* (2004-09).

Eshbaugh-Soha, Matthew. 2010. "The tone of local presidential news coverage." *Political Communication* 27(2):121–140.

Fan, David, David Geddes & Felix Flory. 2013. "The Toyota recall crisis: Media impact on Toyota's corporate brand reputation." *Corporate Reputation Review* 16(2):99–117.

Fogarty, Brian J. 2005. "Determining economic news coverage." *International Journal of Public Opinion Research* 17(2):149–172.

Goidel, Kirby, Stephen Procopio, Dek Terrell & H Denis Wu. 2010. "Sources of economic news and economic expectations." *American Politics Research* .

Goidel, Robert K & Ronald E Langley. 1995. "Media coverage of the economy and aggregate economic evaluations: Uncovering evidence of indirect media effects." *Political Research Quarterly* 48(2):313–328.

Grimmer, Justin & Brandon M Stewart. 2013. "Text as data: The promise and pitfalls of automatic content analysis methods for political texts." *Political Analysis* 21(3):267–297.

Grimmer, Justin, Solomon Messing & Sean J Westwood. 2012. "How words and money cultivate a personal vote: The effect of legislator credit claiming on constituent credit allocation." *American Political Science Review* 106(04):703–719.

Groves, Robert, Floyd Fowler Jr, Mick P. Couper, James M. Lepkowski, Eleanor Singer & Roger Tourangeau. 2009. *Survey Methodology*. 2nd ed. Wiley.

Hastie, Trevor, Robert Tibshirani & Jerome Friedman. 2009. Unsupervised learning. In *The elements of statistical learning*. Springer pp. 485–585.

Hillard, Dustin, Stephen Purpura & John Wilkerson. 2008. "Computer-assisted topic classification for mixed-methods social science research." *Journal of Information Technology & Politics* 4(4):31–46.

Hopkins, Daniel J, Eunji Kim & Soojong Kim. 2017. "Does newspaper coverage influence or reflect public perceptions of the economy?" *Research & Politics* 4(4):2053168017737900.

James, Gareth, Daniela Witten, Trevor Hastie & Robert Tibshirani. 2013. *An introduction to statistical learning*. Vol. 6 Springer.

Jurka, Timothy P, Loren Collingwood, Amber E Boydstun, Emiliano Grossman & Wouter van Atteveldt. 2013. "RTextTools: A supervised learning package for text classification." *The R Journal* 5(1):6–12.

King, Gary, Patrick Lam & Margaret Roberts. 2016. "Computer-Assisted Keyword and Document Set Discovery from Unstructured Text." Working Paper.

Laver, Michael, Kenneth Benoit & John Garry. 2003. "Extracting policy positions from political texts using words as data." *American Political Science Review* 97(02):311–331.

Lyon, Aidan & Eric Pacuit. 2013. The wisdom of crowds: Methods of human judgement aggregation. In *Handbook of human computation*. Springer pp. 599–614.

Mitra, Mandar, Amit Singhal & Chris Buckley. 1998. Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM pp. 206–214.

Monroe, Burt L, Michael P Colaresi & Kevin M Quinn. 2008. "Fightin'words: Lexical feature selection and evaluation for identifying the content of political conflict." *Political Analysis* 16(4):372–403.

Muddiman, Ashley & Natalie Jomini Stroud. 2017. "News values, cognitive biases, and partisan incivility in comment sections." *Journal of Communication* 67(4):586–609.

Page, Scott E. 2008. *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies-New Edition*. Princeton University Press.

Raschka, Sebastian. 2015. *Python Machine Learning*. Packt Publishing Ltd.

Rocchio, Joseph John. 1971. "Relevance feedback in information retrieval.".

Schrodt, Phil. 2011. *Country Infro, 111216.txt*.
URL: *https://github.com/openeventdata/CountryInfo*

Schütze, Hinrich & Jan O Pedersen. 1994. A cooccurrence-based thesaurus and two applications to information retrieval. In *Intelligent Multimedia Information Retrieval Systems and Management-Volume 1*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE pp. 266–274.

Soroka, Stuart N, Dominik A Stecula & Christopher Wlezien. 2015. "It's (Change in) the (Future) Economy, Stupid: Economic Indicators, the Media, and Public Opinion." *American Journal of Political Science* 59(2):457–474.

Stecula, Dominik A & Eric Merkley. 2019. "Framing Climate Change: Economics, Ideology, and Uncertainty in American News Media Content from 1988 to 2014." *Frontiers* *in Communication* 4:6.

Sudman, Seymour, Norman M. Bradburn & Norbert Schwartz. 1995. *Thinking about Answers: The application of Cognitive Processes to Survey Methodology*. Jossey-Bass.

Surowiecki, James. 2005. *The Wisdom of the Crowds*. Anchor.

Tetlock, Paul C. 2007. "Giving Content to Investor Sentiment: The Role of Media in the Stock Market." *The Journal of Finance* 62(3):1139–1168.
URL: *http://dx.doi.org/10.1111/j.1540-6261.2007.01232.x*

Thelwall, Mike, Kevan Buckley, Georgios Paltoglou, Di Cai & Arvid Kappas. 2010. "Sentiment strength detection in short informal text." *Journal of the American Society for Information Science and Technology* 61(12):2544–2558.

Wagner, Christian, Sesia Zhao, Christoph Schneider & Huaping Chen. 2010. The wisdom of reluctant crowds. In *2010 43rd Hawaii International Conference on System Sciences*. IEEE pp. 1–10.

Wu, H Denis, Robert L Stevenson, Hsiao-Chi Chen & Z Nuray Güner. 2002. "The Conditioned Impact of Recession News: A Time-Series Analysis of Economic Communication in the United States, 1987–1996." *International Journal of Public Opinion Research* 14(1):19–36.

Xu, Jinxi & W Bruce Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM pp. 4–11.

Young, Lori & Stuart Soroka. 2012. "Affective news: The automated coding of sentiment in political texts." *Political Communication* 29(2):205–231.