Summarizing large-scale, multiple-document news data: sparse methods and human validation

Luke Miratrix[1] Jinzhu Jia[2]
Brian Gawalt[3]
Bin Yu[1]
Laurent El Ghaoui[3]

[1]Department of Statistics, UC Berkeley, Berkeley, CA 94720
[2]Department of Statistics, Peking University, China
[3]Department of EECS, UC Berkeley, Berkeley, CA 94720
Note: Miratrix and Jia are co-first authors listed in reverse alphabetical order.

May 31, 2011

Abstract

News media plays a significant role in the course of events, political and otherwise. As the amount of news available grows, the task of understanding it grows more difficult for concerned citizens, media analysts, and decision makers alike. In this paper we adapt scalable and sparse statistical techniques to perform a new form of document summarization: subject-specific summarization of multiple news documents. We employ a classification and predictive framework by labeling certain documents as positive examples for a subject and others negative. The predictors are high-dimensional and sparse indicators of words and phrases of a document. The resulted short list of words and phrases for a subject (e.g. "China") is regarded as a summary of the subject relative to a given corpus of documents.

We consider several pre-processing schemes to prepare the text data before applying the sparse predictive framework such as choice of technique to vectorize the documents, or which documents to label and how. We consider the efficacies of four different feature selection approaches—phrase co-occurrence, phrase correlation, L1-regularized logistic regression (L1LR), and L1-regularized linear regression (Lasso)—under these difference pre-processing choices. To evaluate the different summarizers constructed by combining the pre-processing schemes with the four sparse methods, we carry out a carefully designed survey. In the survey, non-expert human readers are asked to rate generated summaries after reading parts of the original documents.

We analyze the human survey results to find that the Lasso is the best overall choice. It consistently produces high-quality summaries across the many pre-processing schemes, sparse feature selection methods, and subject queries. Our findings also reinforce the many years of work suggesting the tf-idf representation is a strong choice of vector space, but only for longer units of text such as articles in newspapers, while L^2 normalization is found to be the best for shorter units of text such as paragraphs.

Even though we focus on print media (newspapers), our framework is general and could be applied to any corpus, even one of considerable size.

Keywords: regularized methods, text summarization, high-dimensional analysis

1 Introduction

The sheer amount and crucial importance of news make it an urgent task to allow efficient summarization. Indeed, the news media significantly drives the course of events. It picks which events

to report and the manner in which to report them, affecting the sentiments of news consumers and through them the wider world. "I am deeply interested," said Joseph Pulitzer in his last will and testament, "in the progress and elevation of journalism, having spent my life in that profession, regarding it as a noble profession and one of unequaled importance for its influence upon the minds and morals of the people."

We know this influence is real. News sources influence how individuals evaluate and elect leaders (Miller and Krosnick, 2000). Exposure to a new source can change how individuals conceive of themselves in relation to the larger world: for instance, Nisbet and Myers (2010) found that greater exposure to the Al Jazeera network reduced nationalistic sentiments among Arab news consumers.

Different news sources produce, for a variety of reasons, different news products. Branton and Dunaway (2009) found that simple geography can introduce coverage bias for some news subjects (specifically, that a publication's distance from the US-Mexico border affects tone on immigration). Gilens and Hertzman (2000) found that coverage of the 1996 Telecommunications Act was considerably different among news sources whose owners stood to gain from the Act's passage versus those news sources without such owners.

If exposure to news can drive larger changes in society, and if news coverage—even when controlling for topic—can vary in tone, emphasis, and style, then it is important to understand precisely where and how these variations occur. It is often said that news media are an essential part of any democracy. In a digital democracy we can provide concerned citizens and decision-makers with automated methods for news summarization so this part can be readily examined.

In this paper we address the problem of efficiently summarizing the way a subject is described in a large collection of text documents. We examine various classification methods that are adapted to the task, and assess their performance based on human validation experiments and a real-life data set of news articles.

1.1 Analyzing news media

Often, news reports are compared to each other by way of hand coding. Wahl, Wood, and Richards (2002) asked volunteer readers to compare their impressions of articles written about mental illness in 1989 versus 1999, with 300 articles drawn from each year. Denham (2004) analyzed the 2003 story of athlete Carl Lewis's possible drug use as covered by United States news sources versus coverage in non-U.S. outlets, examining 115 articles in all. Potter (2009) examined the 2004 news coverage of Haiti by five sources, reading 711 articles in all.

Common to all these approaches is the reduction of each article to a few essential facets, usually established prior to analysis. This approach reduces complex text-data to a handful of quantitive variables, allowing for more traditional analysis such as regression or simple tests of difference. For instance, Wahl et al. (2002) asked coders, "Which of the following themes (e.g., 'mental illness is treatable', 'people with mental illness may be dangerous', etc.) are mentioned in this 1989 article?" Denham coded the types of sources cited in each article. Potter counted the number of times each of a set of keywords (e.g., *"violence," "crisis,"* and *"anarchy,"*) were used in conjunction with *"Haiti."*

Hand-coding is prohibitively labor intensive. Denham relayed in personal correspondence that each article took roughly fifteen minutes to analyze, suggesting about 28 hours of time for their full text corpus. Many interesting studies may never be conducted due to the time and labor expenses involved. Those analyses which are conducted likely undersample the full range of relevant articles, or choose subjects that involve few documents—in all cases, the budget for labor restricts

analysis. As the amount of available news increases, the task of understanding news content grows more daunting. Many are now attempting to help with this via a variety of approaches: Media watchdogs (Media Research Center, Media Matters for America) and automated analogues (Google news trends, Twitter's trending queries) all attempt to make sense of these vast volumes of text. In the last five years we have seen the emergence of a computational social science field connecting statistics and machine learning to anthropology, sociology, public policy, and more (Lazer, Pentland, Adamic, Aral, Barabsi, Brewer, Christakis, Contractor, Fowler, Gutmann, Jebara, King, Macy, Roy, and Van Alstyne, 2009).

In this vein, we believe there is opportunity to answer the question, "What is being said in the news?" with statistical machine learning tools. Given a corpus of documents as well as a subject of interest (represented as a short list of words), our task is to find what few words and phrases best describe the subject as it appears in the documents. We use scalable, reproducible machine learning techniques to reduce corpora of many millions of words into a few representative key-phrases. We view these lists of phrases as summaries of how the given *subject* is *treated* in the corpus.

By *subject*, we mean a noun, topic, or theme of interest in a collection of text documents (e.g., country, person, economy, etc.), and by *treated* we mean how a collection of documents discusses a specific subject and in what context. A *summarizer*, in this case, is an automated process that takes a collection of documents and a subject of interest and returns the summary, i.e. a list of key-phrases describing how the subject is treated across the documents.

To illustrate, one of our proposed summarizers gives *"beijing, contributed research, global, hu jintao, imports, of xinjiang, peoples liberation army, shanghai, sichuan province, staterun, tibet, trade, uighurs, wen jiabao, xinhua news agency"* for how China (represented as *"china, chinas, chinese"*) is treated in the New York Times international section, 2009. This succinct summary captures main relevant personalities (e.g., Wen Jiabao, Hu Jintao), associated countries and areas (e.g., Uighurs, Tibet), entities (Xinhua news), and topics (trade, imports, global [activity], state-run [organizations]). The appearance of these particular aspects of China informs us about how China is being treated by the New York Times, and suggests directions for further human reading.

Even under our general approach, there are many different ways one might design a summarizer. In particular, how raw text is prepared for statistical analysis can have enormous impact on the final results. We therefore investigate how different techniques and approaches differ in the quality of summaries produced. We measure this quality with a human experiment.

Overall, we propose our approach of subject-specific automatic summarization of a large corpus as a method for news media study. This approach readily generalize to other types of documents. We are currently working on making an on-line toolkit, SnapDragon[1], to make these methods readily available.

1.2 Automated analyses

As the news content grows and because of the importance of news contents to society, there is a need to automate the summarization process of written news. Automated and well-validated analyses can increase substantially both the range of queries and the volume of data. In this section we review some existing computational approaches.

[1] http://statnews2.eecs.berkeley.edu/snapdragon

Summarization by extraction. There are two major approaches to text analysis, key-phrase extraction (listing key-phrases for the document) (e.g., Rose, Engel, Cramer, and Cowley, 2010; Senellart and Blondel, 2008; Frank, Paynter, Witten, Gutwin, and Nevill-Manning, 1999; Chen, Yang, Zhang, Chen, Shen, and Cheng, 2006) and sentence extraction (pulling out the "most relevant" sentences to generate a summary) (e.g., Hennig, 2009; Goldstein, Mittal, Carbonell, and Kantrowitz, 2000; Neto, Freitas, and Kaestner, 2002). They involve clipping relevant and exemplary portions of a document as a summary. This line of research has primarily focused on summarizing individual documents, with one summary for every document in a corpus.

When there are many documents, even a short summary of every document adds up quickly and desired content could be buried in a sea of summaries if most documents are not diretly related to the query of interest. If many documents are similar, the collection of summaries becomes redundant. Moreover, if the query of interest is usually mentioned in a secondary capacity it might be missing entirely from the summaries. To address some of these problems, Goldstein et al. (2000) worked on summarizing multiple documents at once to remove redundancy; Hennig (2009) fitted a model to do query-specific summarization of multiple documents. Both these approaches use sentence extraction and do not explicitly use the predictive framework. We use a sparse predictive framwork followed by human experiment to summarize multiple documents. Monroe, Colaresi, and Quinn (2008) demonstrate a technique for finding words of greatest contrast between two classes of text. Their approach groups all text within a class as a single document, while we analyze the corpus of text through individual document units. Our summarizer of a corpus is a list key-phrases pertaining to a specific query of interest.

Summarization via topic modeling. Some analysis algorithms take text information as input and produce a model, usually generative, fit to the data. The model itself captures structure in the data, and this structure can be viewed as a summary. A popular example is the latent Dirichlet allocation (Blei, Ng, and Jordan, 2003), which posits that each word observed in the text is standing in for a hidden, latent "topic" variable. The rates at which words stand in for topics, and which topics appear most frequently, provide a model-based summary: take the most prominent words in a topic.

Chang, Boyd-Graber, Gerrish, Wang, and Blei (2009) had humans evaluate the cohesion of the words representing learned topics. Respondents were asked to identify "imposter" words inserted into lists of words representing a topic. This showed these approaches as producing cogent and reasonable topics: topic words cohered. Supervised versions (Blei and McAuliffe, 2008) of these methods can be used to find features which comparatively distinguish labelled documents.

Although these methods are computationally expensive and produce dense models requiring truncation for interpretability, they are powerful indications of the capabilities of computer-assisted summarization. These methods analyze the corpus as a whole and model how the documents cover a modest number of organically grown topics. We opt instead for a more directed process of specifying a particular subject (out of possible millions) and extracting how that subject is treated in the corpus.

1.3 Our approach: a predictive framework

Our approach is motivated by predictive classification frameworks. Classification of text documents using the words and phrases in those documents as features is a familiar and well-studied prediction problem (Genkin, Lewis, and Madigan, 2007; Zhang and Oles, 2001). In a typical classification

scenario, objects belong to different groups and features of those objects are used to predict that group membership.

We turn subject-specific summarization into a classification problem by forming two groups: subject-related articles and irrelevant. In Section 3.3 to follow, we discuss several plausible ways to label a text unit subject-related or not. For example, for the subject "China, Chinese," we could label all text units with these words in as positive examples and the remainder as negative. We use a predictive classifier to summarize by taking those words and phrases most important for classification as a summary of the subject relative to the corpus.

A predictive framework consists of n units, each with a class label $y_i \in \{-1, +1\}$ and a collection of p possible features that can be used to predict this class label. Each unit $i \in \mathcal{I} \equiv \{1, \ldots, n\}$ is attributed a value x_{ij} for each feature $j \in \mathcal{J} \equiv \{1, \ldots, p\}$. These x_{ij} form a $n \times p$ matrix X. The n units are blocks of text taken from the corpus (e.g., entire articles or individual paragraphs), the class labels y_i indicate whether document unit i contains content on a subject of interest, and the features are all the possible key-phrases that could be used to summarize the subject. y and X can be built in several ways. We build X by reweighting the elements of a document-term matrix C:

Definition A document-term matrix C sets

$$C_{ij} := \text{ The number of times key-phrase } j \text{ appears in document } i$$

This is often called the *bag-of-phrases model*: each document is represented as a vector with the jth element being the total number of times that the specific phrase j appears in the document. Stack these row vectors to make a matrix $C \in \mathbb{R}^{n \times p}$ of counts. C has one row for each document and one column for each phrase. C tends to be highly sparse: most entries are 0.

C and X can be viewed as numerical (e.g., Hennig, 2009; Goldstein et al., 2000; Neto et al., 2002) vector representations of the documents. Each document is a point in $\mathbb{R}^p$, with each dimension corresponding to a feature.

To transform raw text into this vector space, convert it to a collection of individual text document units, establish a dictionary of possible phrases, and count how often each of the dictionary's phrases appear in each of the document units. Once this is completed, the summarizing process consists of 3 major steps:

1. build X from C;
2. build y by identifying which document units in the corpus treat a specified subject;
3. extract a list of phrases which summarize the documents that treat the subject (compared to those that do not).

The overall arc is illustrated in Figure 1. Each step allows for specific choices. We describe these steps in detail below and introduce the notation and terminology we use to discuss them. Following that, we discuss the results of the human experiment in which we investigate which choices lead to better summaries, and identify overall methods that are stable under minor changes of implementation.

Establishing the predictive framework for text summarization requires a number of components: how the documents are *labelled*, how the documents are *vectorized*, and how phrases are *selected*. These can all be done in different ways. In this paper, we build multiple summarizers from different combinations of these components and compare their relative effectiveness by asking human readers

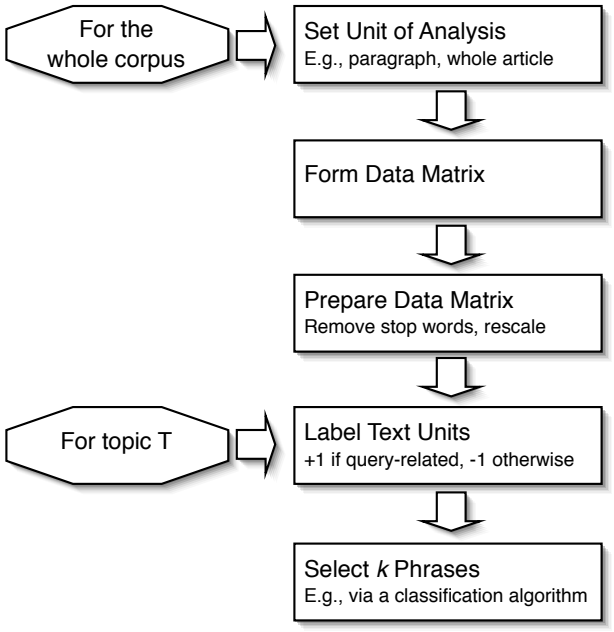

Figure 1: The Summarization Process. The first two steps are done for the entire corpus. The latter steps are done for a specific subject. There are multiple choices an experimenter can make for each box. These choices would result in different summarizers, some better than others. All steps explained in text.

to register their approval or disapproval of the resulting summaries for several subjects of interest in a randomized experiment. The different choices for the components of the summarizers are described fully in Sections 3 and 4.

1.4 Evaluating summaries

Different combinations of choices for the vectorization, labeling, and feature selection steps produce different summarizers. Four summaries of four different countries from one set of choices are shown in Table 1.3. These summaries arguably serve as an insight into the New York Times' coverage of these countries, and provide some pointers as to future directions of more in-depth analysis. But are these summaries better, or worse, than the summaries from a different configuration?

Comparing the efficacy of different summarizers requires systematic evaluation. To do this, many researchers use corpora with existing summaries (e.g., using the human-encoded key-phrases in academic journals such as in Frank et al. (1999)), or corpora that already have human-generated summaries (such as the TIPSTER dataset used in Neto et al. (2002)). Others have humans generate summaries for a sample of individual documents and compare the summarizer's output to the human baseline for that sample of documents. We, however, investigate summarization of many documents in cases where the labeling is ambiguous or non-obvious, and so we do not use an annotated evaluation corpus or summaries of individual documents. The news dataset we use, a collection of New York Times articles from the international section, is described in Section 2 below.

In the machine-learning world, numerical measures such as prediction accuracy or model fit are

Iraq	Russia	Germany	Mexico
american	a medvedev	angela merkel	and border protection
and afghanistan	caucasus	berlin	antonio betancourt
baghdad	europe	chancellor angela	cancn
brigade	gas	european	chihuahua
combat	georgia	france and	denise grady
gen	interfax news agency	frankfurt	drug cartels
in afghanistan	iran	group of mostly	guadalajara
invasion	moscow	hamburg	influenza
nuri	nuclear	marwa alsherbini	oaxaca
pentagon	president dmitri	matchfixing	outbreak
saddam	republics	minister karltheodor zu	president felipe
sergeant	sergei	munich	sinaloa
sunni	soviet	nazi	swine
troops	vladimir	world war	texas
war and who			tijuana

Table 1: Four Sample Summaries of Four Different Countries. The method used was one of the best identified for article-unit analysis by our validation experiment: a count rule with a threshold of 2, the Lasso for phrase selection, and tf-idf reweighting of features. These summaries inform us as to which aspects of these countries are of most concern to the New York Times in 2009: even now, Nazis and the World Wars are tied to Germany. Iraq and Afghanistan are also tied closely. Gen[erals] and combat are the major focus in Iraq. The coverage of Mexico revolves around the swine flu, drug cartels, and concerns about the border. Russia, on the other hand, does not seem strongly associated with any particular event. These observations may well be unique to the New York Times. It would be instructive to compare to other media channels.

often used to compare different techniques. While we hypothesize that prediction accuracy should correlate with summary quality, there are no theoretical results to demonstrate this. Furthermore, there are no other immediate calculable measures of summary quality, and thus evaluating summarizer performance with numerical measures is not be robust to critique.

We therefore asked non-expert human readers to compare samples of news articles of interest to samples of relevant summaries generated by our different proposed methods for a variety of subjects. We then compared these ratings across summarization method to discern which methods were most satisfactory. This survey is described in Section 5.

The results, discussed in Section 6, show that the Lasso, with the robustness to produce high-quality summaries across the many pre-processing schemes, is the best choice of feature selection engine. Its cousin, L^1 regularized logistic regression, performs comparably well but at greater computational expense. Our findings also reinforce the many years of work suggesting the tf-idf representation is a strong choice of vector space.

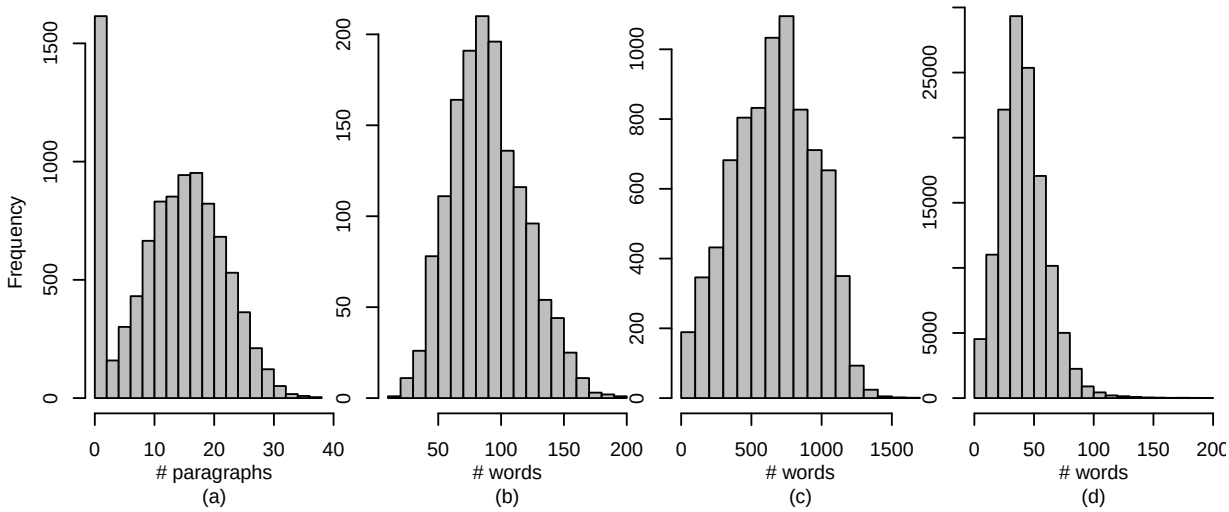

Figure 2: Length Distribution of the New York Times dataset. Left plot, (a), is the distribution of article lengths in paragraphs. The right three plots show different distributions of lengths in words: (b) is length of single-paragraph articles, (c) is length of longer articles, and (d) is length of the paragraphs of the longer articles (the chart is truncated at 200 words—there are 18 paragraphs with more than 200 words).

2 Description of Text Corpus

For our investigation we used the International Section of the New York Times for the 2009 year. Articles were scraped from the newspaper's RSS feed[2], and the HTML markup was stripped from the text. We obtained 130,266 paragraphs of text comprising 9,560 articles. The New York Times, upon occasion, will edit an article and repost it under a different headline and link; these multiple versions of the articles remain in the data set. By looking for similar articles as measured by high values of the cosine of the angle between their respective feature vectors, we estimate that around 400 articles (4–5%) have near-duplicates.

The number of paragraphs in an article ranges from 1 to 38. Typical articles[3] have about 16 paragraphs (with an Inter-Quartile Range (IQR) of 11 to 20 paragraphs, i.e. about 75% of articles are 11 paragraphs or longer, and 25% are 20 paragraphs or longer). However, about 15% of the articles, the "World Briefing" articles, are a special variety that contain only one long paragraph[4]. Among the more typical, non-"World Briefing" articles, the distribution of article length as number of paragraphs is bell-shaped and unimodal. Figure 2 illustrates the distribution of number of paragraphs per article, and number of words per paragraph and per article (broken down by article type).

The single-paragraph "World Briefing" articles have a median length of 87 words. For the longer articles, the median number of words in an article is 664 and the median number of words in a paragraph is 38 words. Distributions of these three statistics are shown on the right three plots

[2] `feed://feeds.nytimes.com/nyt/rss/World`

[3] See, e.g., `http://www.nytimes.com/2011/03/ 04/world/americas/04mexico.html`

[4] See, e.g., `http://www.nytimes.com/2011/03/03/ world/americas/03briefs-cuba.html`

of Figure 2. Longer articles have much shorter paragraphs, generally, than the "Word Briefing" single-paragraph articles.

3 Building the Predictive Framework

3.1 Choosing the granularity of analysis

We divide the raw text into units of analysis and determine which of those units have relevant information about the subject, and which do not, and summarize based on common features found in these units. The granularity with which the text is partitioned may then have some impact on the resulting summaries. In particular, we hypothesized that using smaller, lower-word-count units of text should produce more detail-oriented summaries, while using larger units will highlight key-phrases dealing more with the larger themes discussed when the subject of interest is mentioned.

We tested this hypothesis by comparing summarizers that analyze at the article level to those which analyze at the component-paragraphs level. Interestingly, we found no large differences. See Section 6.4.

3.2 Identifying potential summarizing phrases

To build C we must first identify all possible phrases that could be part of a summary. This list of possibilities constitute our *dictionary*. Building this dictionary begins with asking, "Which text phrases are acceptably descriptive?" Sometimes the answer to this question suggests a manually-defined dictionary: if summaries should only list, e.g., countries then the dictionary would be easy to assemble by hand in advance of seeing the corpus.

In many situations, however, the dictionary of terms should be kept large. This means automatically drawing the dictionary from the corpus itself. Different decisions—Should capitalization matter? Are punctuation marks terms? Can terms include numerals?—yield dictionaries varying widely in size and utility. Terms can be further identified and distinguished from each other by many natural language tools, e.g. part-of-speech tagging. Any automated system for forming a dictionary will entail at least some uncertainty in term identification.

We elected to use a large dictionary, identifying all phrases of up to three words in length. We generated our dictionary by first removing all numerals and punctuation, then case-folding (i.e., converted all text to lowercase). We then segmented each document into overlapping phrases, consisting of all single words, bigrams and trigrams in that document unit. Some text analysts stem their phrases, reducing words to a core root prefix, e.g., truncating *"terror," "terrorist,"* and *"terrorism"* to *"terror"*. We do not stem. There is a semantic difference if a particular subject is associated with *"canadians"*, the citizenry versus *"canada"* the country. Stemming would lose the ability to make that inferrence. From our corpus, this identified in 4.5 million distinct phrase tokens. We then did a first-pass pruning, removing all phrases appearing fewer than six times in the corpus, resulting in a dictionary of $p = 216,626$ different phrases.

3.3 Labeling the text

Under the predictive framework we train a classifier to predict document labels, $y_i \in \{-1, +1\}$, with their vectors of phrase features, $x_i \in \mathbb{R}^p$, for $i = 1, \ldots, n$. In the labeling step we build $y = (y_1, \ldots, y_n)$ by deciding whether each text unit in the corpus should be considered a positive

class example or a negative class example. Establishing the class labels for a news document corpus is sometimes straightforward. For instance, when Wahl et. al compared 1989 articles about mental illness to those from 1999, the labels are simple: the documents from the opposing years go in opposite classes. We build y by identifying which of the document units treat the subject of interest. For small enough n, y could be built by hand. For corpora too large to admit manual labeling, we need reasonable automatic labeling. Ideally this need not be a perfect identification—noise in labeling should not have undue impact on the resulting summaries.

In the large sense, a subject is a concept of interest that an investigator might have. We represent a subject with a small set of phrases, e.g., the subject of China would be well represented by the set {*"china," "chinas," "chinese"*}. Specifically, let the *subject* $Q \subset \mathcal{J}$ be a set of phrases selected to ideally capture a concept of interest.

count-m. We consider two general labeling techniques. The first technique, count-m, marks text unit i as treating a subject if related phrases appear frequently enough, as given by:

Definition Count-m labeling labels text unit i as:

$$y_i = 2 \cdot \mathbb{1}\{r_i \geq m\} - 1$$

where $\mathbb{1}\{\cdot\}$ is the indicator function and $r_i \equiv \sum_{j \in Q} c_{ij}$ is the total number of subject-specific phrases in unit i.

hardcount-m. The second labeling technique, hardcount-m, drops any document i with $0 < r_i < m$ from the data set instead of labeling it with -1. The hardcount method considers those documents too ambiguous to be useful as negative class examples. It produces the same positive example set as count-m would. We hypothesized that dropping the ambiguous document units will heighten the contrast in content between the two classes, and thus lead to superior summaries. It did not. See Section 6.4.

In conjunction with labeling the documents, we also remove the columns corresponding to each phrase in subject Q from the set of possible features $\mathcal{J}$ to prevent the summary from being trivial and circular: appearance of the phrase *"united states"* perfectly distinguishes the set of documents containing the phrase *"united states"*. We also remove sub-phrases and super-phrases. For example, if Q is {*"united states"*} then candidate summary phrases *"united"*, *"states"*, *"of the united"*, and *"states of america"* would all be removed. The removal is easily automated. For ease of notation in the following text, when we state X, we mean the X for which this removal has already been performed for the relevant subject.

We generate y this way because we are interested in how a subject is treated in a corpus. Other approaches are possible. For example, y might identify which articles were written in a specific date range; this would then lead to a summary of what was covered in that date range.

3.4 Reweighting and removing features

It is well known that baseline word frequencies impact information retrieval methods, and so often raw counts are adjusted to account for commonality and rarity of terms (e.g., Monroe et al., 2008). In the predictive framework, this adjustment is done in the construction of the feature matrix X. We first transform each text unit into a vector of features where the features are short phrases of

up to three words long. We consider three different constructions of X, or vectorizations of the text, all built on the bag-of-tokens representation.

Our first, baseline, vectorization is to simply drop stop words (words a priori determined as too uninformative to merit inclusion). Our second approach is to rescale each token vector (column of C) to have unity L^2 norm. Our third is an implementation of the tf-idf technique (Salton, 1991), rescaling the bag-of-tokens components so that appearances of rarer tokens are considered more important than common ones.

Stop Words. Stop words are low information words such as *"and,"* or *"the"*, typically appearing with high frequency. Stop words may be context dependent. For example, in US international news *"united states"* or *"country"* might be considered high frequency and low information. High-frequency words have higher variance and effective weight in many methods, causing them to be erroneously selected as features due to sample noise. To deal with these nuisance words, many text-processing methods use a fixed, hand-built stop-word list and preemptively remove all features on that list from consideration (e.g., Zhang and Oles, 2001; Ifrim, Bakir, and Weikum, 2008; Genkin et al., 2007).

This somewhat ad-hoc method does not adapt automatically to the individual character of a given corpus and presents many difficulties. Switching to a corpus of a different language would require new stop word lists for the summarizer. When considering phrases instead of single words, the stop word list is not naturally or easily extended. For example, simply dropping phrases containing any stop word is problematic: it would be a mistake to label *"shock* and *awe"* uninteresting. On the other hand, there are very common candidate phrases that are entirely made up of stop words, e.g., *"of the,"* so just culling the single word phrases is unlikely to be sufficient. See Monroe et al. (2008) for further critique.

L^2-rescaling. As an alternative, appropriately adjusting the document vectors can act in lieu of a stop-word list by reducing the variance and weight of the high-frequency features. We use the corpus to estimate baseline appearance rates for each feature and then adjust the matrix C by a function of these rates; this core idea is discussed by Monroe et al. (2008).

We define L^2-rescaling to be:

Definition X is a *L^2-rescaled* version of C if each column of C is rescaled to have unit length under the L^2 norm. I.e.:

$$x_{ij} = \frac{c_{ij}}{\sqrt{z_j}}, \text{ where } z_j \equiv \sum_{i=1}^{n} c_{ij}^2$$

To illustrate, see Table 2, containing four summaries from the L1LR feature selection method. Column 1 is with stop-word removal only and Column 2 is with both stop-word removal and L^2-rescaling. Despite stop-word removal, the first list contains some meaningless words such as *"mr",* while the rescaled list does not contain these words. Indeed, without rescaling, *"mr"* and *"said"* appear for most subjects examined. Furthermore, the remaining words are quite general. This is a common problem with stop-word lists: they get rid of the worst offenders, but do not solve the overall problem. If no stop words are deleted, then we get column 3 (no rescaling or stop-word removal) and column 4 (rescaling only). Column 3 is terrible; the list is dominated by high-frequency, low-content words. Column 4 is identical to Column 2—the rescaling, in this case, has rendered the stop-word list irrelevant.

	stop-word only	stop word and rescaling	no adjustment	rescaling only
1	afghanistan	asian	afghanistan	asian
2	beijing	beijing	and	beijing
3	companies	contributed research	beijing	contributed research
4	countries	euna lee	countries	euna lee
5	economic	global	global	global
6	global	hong kong	has	hong kong
7	hong	jintao	his	jintao
8	military	north korea	its	north korea
9	mr	shanghai	mr	shanghai
10	north	staterun	north	staterun
11	percent	uighurs	of	uighurs
12	the united states	wen jiabao	the united	wen jiabao
13	uighurs	xinhua	to	xinhua
14	world		united states	
15	year		was	

Table 2: Comparative Effects of Reweighting Methods. Four summaries of "China" from all combinations of L^2-rescaling and stop-word removal. The phrase-selection method used is L1LR with count-2 labeling on full articles.

tf-idf Weighting. An alternate rescaling comes from the popular tf-idf heuristic (Salton, 1991), which attempts to de-emphasize commonly occurring terms while also trying to account for each document's length.

Definition X is a *tf-idf weighted* version of C if

$$x_{ij} := \frac{c_{ij}}{q_i} \log\left(\frac{n}{d_j}\right)$$

where $q_i \equiv \sum_{j=1}^{p} c_{ij}$ is the sum of the counts of all key-phrases in document i and $d_j \equiv \sum_{i=1}^{n} \mathbb{1}\{c_{ij} > 0\}$ is the number of documents in which term j appears at least once.

Under tf-idf, words which appear in a large proportion of documents are shrunk considerably in their representation in X. Words which appear in all n documents, such as *"the"*, are zeroed out entirely. A potential advantage of tf-idf is that it might ease comparisons between documents of different lengths because term counts are rescaled by the total count of terms in the document.

We hypothesized that feature weighting is more transparent and reproducible than stop-word removal and that it results in superior summaries when compared to stop-word removal. With the human validation experiment, we compared using L^2-rescaling, tf-idf weighting, and stop-word removal as the pre-processing step for each of our feature selectors and found that humans indeed prefer lists coming from reweighting methods.

4 Feature selection

Classic prediction methods give models where each feature usually is given a weight. The models are thus hard to interpret when there are many features, as is typically the case with text analysis (our data set contains more than 4,500,000 potential phrases). We, however, want to ensure that the number of phrases selected is small so the researcher can easily read and consider the entire summary. Short summaries are quick to digest, and thus are easily comparable. Such summaries might even be automatically generated for a corpus in one language and then translated to another, thus easing comparison of media coverage from different nationalities and allowing insight into foreign language news. Fundamentally, a small set of features is more easily evaluated by human researchers.

The constraint of short summaries makes the summarization problem a sparse feature selection problem, as studied in, (e.g., Forman, 2003; Lee and Chen, 2006; Yang and Pendersen, 1997). *Sparse* methods, such as L^1-penalized regression, naturally select a small subset of the available features (in our case candidate key-phrases) as being relevant predictors. Sparse methods can be computationally intensive, but in order to easily explore a corpus, feature selection needs to be quick; we discuss relative speeds below.

In other domains, L^1-regularized methods are useful for sparse model selection; they can identify which of a large set of mostly irrelevant features are associated with some outcome. In our domain there is no reasonable underlying model that is indeed sparse; we expect different phrases to be more or less relevant, but few to be completely and utterly irrelevant. Nevertheless, we still employ the sparse methods to take advantage of their feature selection aspects, hoping that the most important features will be selected first.

Given the features X and document labels y for a subject, we extract the columns of X that constitute the final summary four different ways. Two of them, Co-occurrance and Correlation Screening, are scoring schemes where each feature gets scored independently and the top-scoring features are taken as a summary. This is similar to traditional key-phrase extraction techniques. The other two (the Lasso and L1LR) are L^1 regularized version of least squares linear regression and logistic regression, respectively. As an example of different summaries of the same subject, Table 3 has four summaries for China. These different summaries, which used the same reweighting and labeling methods, demonstrate that choice of feature selector matters greatly.

The *feature selection step* is a regression (or similar) analysis of the vectorized, labeled text. We seek a subset of phrases $\mathcal{K} \subseteq \mathcal{J}$ with cardinality as close as possible to, but no larger than, a target k, the desired summary length. We typically use $k = 15$ phrases, but 30 or 50 might also be desirable. The higher the value of k, the more detailed and complex the summary.

The two simple phrase feature selectors (Co-occurrence and Correlation Screening) rate each feature independently of the others. The two more computationally expensive feature selection methods (the Lasso and L^1-regularized logistic regression) use sparse regression techniques that incorporate relationships between features.

For the two simpler methods, we score all candidate phrases individually and then take the k highest-scoring, *distinct* phrases as the summary. By *distinct*, we mean that we drop all selected sub-phrases when counting list length. For example, if *"united states"* and *"united"* are both selected, we drop *"united"*.

The primary advantages of Co-Occurance and Correlation Screening is that they are fast, scalable, and easily distributed across multiple platforms for parallel processing. Unfortunately, as they score each feature independently from the others, they cannot take advantage of any structure

	Co-Occur	Correlation	L1LR	Lasso
1	and	beijing and	asian	asian
2	by	beijings	beijing	beijing
3	contributed research	contributed research	contributed research	contributed research
4	for	from beijing	euna lee	exports
5	global	global	global	global
6	has	in beijing	hong kong	hong kong
7	hu jintao	li	jintao	jintao
8	in beijing	minister wen jiabao	north korea	north korea
9	its	president hu jintao	shanghai	shanghai
10	of	prime minister wen	staterun	tibet
11	that	shanghai	uighurs	uighurs
12	the	the beijing	wen jiabao	wen jiabao
13	to	tibet	xinhua	xinhua
14	xinhua	xinhua the		
15	year	zhang		

Table 3: Comparison of the Four Feature Selection Methods. Four sample summaries of news coverage of China. (Documents labeled via count-2 on articles, X from L^2-rescaling.) Note superior summary quality of the right two feature selectors that explicitly use the predictive framework. See text for details.

between features to aid summarization.

For the two sparse regression methods, we tune the usual regularization parameters to achieve k non-zero, distinct phrases and then take these resulting phrases as our summary. Sparse methods typically incur a heavier computational burden, though this extra computational cost would be worth paying in order to select a more representative and less redundant set of features.

It is important to underline that, in our case, the object of interest has shifted: we use classification methods but are not interested in classification. Our underlying hypothesis is that those features useful for classification are the very features that humans would judge as viable and accurate summaries of the subject being classified in the corpus given. We show in the human validation experiment that this is indeed the case.

We detail these four specific phrase-selection methods below. Two of these, Co-occur and L1-penalized logistic regression (L1LR), are familiar schemes from previous work Gawalt, Jia, Miratrix, Ghaoui, Yu, and Clavier (2010).

4.1 Co-Occurrence

This is our simplest, baseline, method. The idea is to simply take those phrases that appear most often (or have greatest weight) in the positively marked text as the summary. This method is often used in, e.g., newspaper charts showing the trends of major words over a year (such as Google News Trends[5]) or word or tag clouds (created at sites such as Wordle[6]).

[5] `http://www.google.com/trends`
[6] `http://www.wordle.net/`

By the feature selection step we have two labelled document subsets, $\mathcal{I}^+ = \{i \in \mathcal{I} | y_i = +1\}$, of cardinality $\#\mathcal{I}^+$, and $\mathcal{I}^- = \{i \in \mathcal{I} | y_i = -1\}$, of cardinality $\#\mathcal{I}^-$. Define $\hat{E}(x_j|y{=}{+}1)$ as the mean of x_{ij} over all i such that $y_i = +1$. Also define, e.g., $P(y{=}{+}1)$ as the proportion of y_i that are $+1$. Note that $\#\mathcal{I}^+ = nP(y{=}{+}1)$. Compute the relevance score s_j of feature j for all $j \in \mathcal{J}$:

$$s_j = \frac{1}{\#\mathcal{I}^+} \sum_{i \in I^+} x_{ij} = \hat{E}(x_j|y{=}{+}1).$$

s_j is the expected weight of the phrase in the positively marked examples.

For some k', let $\bar{s}$ be the $(k'+1)$th highest value found in the set $\{s_j \mid j \in \mathcal{J}\}$. Build $K = \{j \in \mathcal{J} \ : \ s_j > \bar{s}\}$, the set of (up to) k' phrases with the highest average weight across the positive examples. Any phrases tied with the $(k'+1)$th highest value are dropped, sometimes giving a list shorter than k'. The size of K after subphrases are removed can be even less. Let the initial value of k' be k, the actual desired length. Now adjust k' upwards until just before the summary of *distinct* phrases is longer than k. We are then taking the $k' \geq k$ top phrases and removing the sub-phrases to produce k or fewer distinct phrases in the final summary.

If $X = C$, i.e. it is not weighted, then s_j is the average number of times feature j appears in $\mathcal{I}^+$, and this method selects those phrases that appear most frequently in the positive examples. The weighting step may, however, reduce the co-occurrence score for common words that appear frequently in both the positive and negative examples. This is especially true if, as is usually the case, there are many more negative examples than positive ones. This type of simple adjustment can radically increase this method's performance.

4.2 Correlation Screening

Correlation Screening selects features with the largest absolute correlation with the subject labeling y. It is a fast method that independently selects phrases that tend to appear in the positively marked text and not in the negatively marked text. Score each feature as:

$$s_j = |cor(x_j, y)|$$

Now select the k highest-scoring, distinct features as described for Co-Occur, above.

As a motivation for this technique, suppose that the features $x_j, j = 1, \ldots p$ have been rescaled to have a variance of 1. L^2 rescaling approximates this, but as the columns are not 0-mean, the variances are going to be less than 1. The empirical covariance $cov(x_j, y) = s_j \cdot \sqrt{var(x_j)var(y)} = s_j \cdot \sqrt{var(y)}$ of x_j and y is then:

$$cov(x_j, y) \quad \propto \quad \hat{E}(x_j|y{=}{+}1) - \hat{E}(x_j|y{=}{-}1)$$

The proportionality constant is fixed by y and constant across all features. To derive the above, expand the expectations around the two values of y, and use the identity $P(y{=}{+}1) + P(y{=}{-}1) = 1$.

Co-Occur gives scores proportional to $\hat{E}(x_j|y = 1)$ only. A high score for feature j does not necessarily indicate any connection to the target y, and could instead be obtained if j is an overall common phrase. Correlation Screening does more: the difference between $\hat{E}(x_j|y{=}{+}1)$ and $\hat{E}(x_j|y{=}{-}1)$ must relate to the connection between x_j and y. Due to the fact that we do not recenter the x_j, the above argument only completely holds when comparing features with similar variances, which are basically functions of frequency of appearance. Not being centered penalizes features with high variance, which has been generally seen as a good benefit (see, e.g., Monroe et al., 2008).

4.3 L1-penalized linear regression (Lasso)

The Lasso is an L^1-penalized version of linear regression and is the first of two feature selection methods examined in this paper that address our model-sparsity-for-interpretability constraint explicitly. Imposing an L^1 penalty on a least-squares problem regularizes the vector of coefficients, allowing for optimal model fit in high-dimensional ($p > n$) regression settings. Furthermore, L^1 penalties typically result in sparse feature-vectors, which is desirable in our context. The Lasso takes advantage of the correlation structure of the features to, in principle, avoid selecting correlated terms. For an overview of the Lasso and other sparse methods see, e.g., *The Elements of Statistical Learning* (Hastie, Tibshirani, and Friedman, 2003).

The Lasso is defined as:

$$(\hat{\beta}(\lambda), \hat{\gamma}) := \arg \min_{\beta,\gamma} \sum_{i=1}^{m} \left\| y - x_i^T \beta - \gamma \right\|^2 + \lambda \sum_j |\beta_j|. \tag{1}$$

The penalty term λ governs the number of non-zero elements of β. We use a non-penalized intercept, γ, in our model. Penalizing the intercept would shrink the estimated ratio of number of positive example documents to the number of negative example documents to 1. This is not desirable; the number of positive examples is far less than 50%, as shown in Table 6, and in any case is not a parameter which needs estimation for our summaries. We solve this convex optimization problem with a modified version of the BBR algorithm (Genkin et al., 2007) described further in Section 4.5.

4.4 L1-penalized logistic regression (L1LR)

Similar to the Lasso, L1-penalized logistic regression (L1LR) is typically used to obtain a sparse feature set for predicting the log-odds of an outcome variable being either $+1$ or -1. It is widely studied in the classification literature, including text classification (see Genkin et al., 2007; Ifrim et al., 2008; Zhang and Oles, 2001). We define the model as:

$$(\hat{\beta}(\lambda), \hat{\gamma}) := \arg \min_{\beta,\gamma} - \sum_{i=1}^{m} \log \left(1 + \exp[-y_i(x_i^T \beta + \gamma)]\right) + \lambda \sum_j |\beta_j|. \tag{2}$$

The penalty term λ again governs the number of non-zero elements of β. As with Lasso, we again do not penalize the intercept. We implement L1LR with a modified form of BBR.

4.5 Implementation and Computational cost

Computational costs primarily depend on the size and sparsity of X. We store X as a list of tuples, each tuple being a row and column index and value of a non-zero element. This list is sorted so it is quick to identify all elements in a matrix column. This data structure saves both in storage and computational cost.

Let Z be the number of nonzero elements in X. Then the complexity for the tf-idf and L^2 rescaling methods are $O(Z)$ because we only have to re-weight the nonzero elements and the weights are only calculated from the nonzero elements. Stop-word elimination is also $O(Z)$.

The running times of the four feature selection methods differ widely. For Correlation Screening and Co-ocurrence, the complexity is $O(Z)$.

The Lasso and L1LR depend on solving complex optimization problems. We implemented them using a modified form of the BBR algorithm (Genkin et al., 2007). The BBR algorithm is a coordinate descent algorithm to solve L^1 penalized logistic regressions with penalized (or no) intercept. It cycles through all the columns of X, computing the optimal value for β_j for feature j holding the other features fixed. The intercept would simply be another column in X with all 1s.

We modified the BBR algorithm such that 1) we can solve the Lasso with it; 2) we do not penalize the intercept; and 3) the implementation exploits the sparsity of X. Not penalizing the intercept preserves sparsity, even if we chose to center the columns of X. For both the Lasso and Logistic regression, we need only a few iterations through β_1 to β_p to get the final solution.

For each iteration, we first calculate the optimum intercept $\hat{\gamma}$ given the current value of $\hat{\beta}$ and then cycle through the features, calculating the update to $beta_j$ of $\Delta\beta_j$ for $j = 1, \ldots, p$. The complexity for calculating $\Delta\beta_j$ is $O(Z_j)$, where Z_j is the number of nonzero elements in the jth column of X because calculation of $\Delta\beta_j$ involves only a few simple mathematical operations $(+, -, *, /)$ on the non-zero elements of X's jth column. Given this, we have a complexity cost of $O(Z)$ for each full cycle through the features. The overall computational complexity of the Lasso and Logistic regression is then $O(Itr \times K)$ where Itr is the number of iterations needed.

For both the Lasso and L1LR, higher values of λ result in the selection of fewer features. A sufficiently high λ will return a β with zero weight for all phrases, selecting no phrase features, and $\lambda = 0$ reverts the problem to ordinary linear regression, leading to some weight put on all phrases in most circumstances. By doing a binary search between these two extremes, we quickly find a value of λ for which $\beta(\lambda)$ has the desired k distinct phrases with non-zero weight. The total computational cost is thus $O(Rnd \times Itr \times K)$, with Rnd the number of steps needed in this binary search.

When not exploiting the sparse matrix, the complexity for a single optimization step is $O(Itr \times n \times p)$. When $K \ll n \times p$, our implementation saves a lot of computation cost.

Empirical Speed Tests. We timed the various methods to compare them given our data set. The average times to summarize a given subject for each method, not including the time to load, label, and rescale the data, are on Table 4. As expected, Co-occurence and Correlation Screening are roughly the same speed. The data-preperation steps by themselves (not including loading the data into memory) average a total of 15 seconds, more expensive by far than the feature selection for the simple methods (although we did not fully optimize these steps). Cost here is primarily due to generating the labeling y and dropping the subject-related features from the matrix X.

Lasso is currently about 9 times slower and L1LR is more than 100 times slower than the baseline Co-occur using current optimization techniques, but these techniques are evolving fast. For example, one current area of research, safe feature elimination, allows for safely pruning many irrelevant features before fitting which can lead to substantial speed-ups (El Ghaoui, Viallon, and Rabbani, 2011). This pre-processing step is computationally very cheap and allows for a huge reduction in the number of features when the penalty parameter is high, which is precisely the regime where the desired list length is short, i.e. fewer than a hundred.

We implemented the sparse regression algorithms and feature correlation calculations in C; the overall package is in matlab. While further optimization is quite plausible, it is clear that L1LR is very slow. The speed cost for Lasso, especially when considering the overhead of labeling and rescaling, is fairly minor, as shown on the third column of Table 4.

	Phrase selection (sec)	Total time (sec)	Percent increase
Co-occur	1.0	20.3	
Correlation Screen	1.0	20.3	0%
The Lasso	9.3	28.7	+41%
L1LR	104.9	124.2	+511%

Table 4: Computational Speed Chart. Average running times for the four feature selection methods over all subjects considered. Second column includes time to generate y and adjust X. Final column is percentage increase in total time over co-occur, the baseline method.

4.6 The impact of selecting distinct phrases

Final summaries consist of a target of k *distinct* key-phrases. The feature-selectors are adjusted to provide enough phrases such that once sub-phrases (e.g., *"united"* in *"united states"*) are removed, the list is k phrases long. This removal step, similar to stop-word removal, is somewhat ad hoc. It would be preferable to have methods that naturally select distinct phrases that do not substantially overlap. Sparse methods have some protection against selecting highly correlated features, and thus they might not need this cleaning step as sub-phrases tend to be highly correlated with parent phrases, with correlations often exceeding 0.8. To investigate this, we examined the average value of $k' - k$, the difference of the length of the summary without sub-phrases removed to the length with this removal. Results are shown in Table 5. The sparse methods indeed do not need to take advantage of this step, supporting the heuristic knowledge that L^1-penalization tends to avoid selecting correlated features. Under tf-idf, only a little over 1 phrase, on average, is dropped. The independent feature selection methods, however, drop many phrases on average.

	Reweighting Method		
Feat. Sel. Method	stop-word	L^2-rescaling	tf-idf rescaling
Co-Occur	2.7	12.8	7.3
Correlation	12.9	12.9	12.7
L1LR	0.5	3.9	1.2
Lasso	0.6	3.7	1.2

Table 5: Phrase Reduction for the Four Feature Selectors. Each entry shows the mean number of sub-phrases dropped, on average, for all varieties of summarizer with specified rescaling and feature-selection method for a target summary length of $k = 15$ phrases. For example, under tf-idf we need to generate a full list of 16.2 phrases with L1LR, on average, to achieve a final list length of 15 phrases. The sparse methods do not need much pruning. Correlation Screening, as anticipated, selects highly related sub-phrases and therefore requires much pruning.

5 Human Experiment

A summarizer's resulting phrase list should be short and interpretable, and should reflect how humans themselves, had they the time or ability to read the corpus, would summarize the subject. The list should be comprehensive, give informative words, and contain few redundancies. It ideally should touch on a broad range of aspects of the subject, suggesting a premium be placed on how unrelated a summary's phrases are from each other. Many techniques might give short lists of phrases potentially related to a given subject. The difficulty lies in how to evaluate these different techniques in terms of relevant metrics—salience, accuracy, and utility. Classical measures of performance such as prediction accuracy might indicate these things—indeed, this belief motivates our overall approach—but we do not know the strength of the connection.

Although time consuming, people can tell how well a summary relates to a subject, as the hand-coding practice in media analysis shows. Because final outcomes of interest are governed by human opinion, the only way to validate that a summarizer is achieving its purpose is via a study where humans assess summary quality. We therefore design and conduct such a study. Our study has three main aims: to verify that features used for classification are indeed good key-phrases, to help learn what aspects of the summarizers seem most important in extracting the key meaning of a corpus, and to determine which feature selection methods are most robust to different choices of pre-processing (choice of granularity, labeling of text units, and rescaling of X).

We compare our four feature selection methods under a variety of labeling and vector-reweighting methods in a crossed, randomized experiment where non-experts read both original documents and our summaries and judge the quality and relevance of the output. Even though we expect individuals' judgements to vary, we can average the responses across a collection of respondents and thus get a measure of overall, generally shared opinion.

We carried out our survey in conjunction with the XLab, a campus lab dedicated to helping researchers conduct human experiments. The lab provides a room of kiosks where researchers can ask a series of questions of their respondents in a focused, controlled environment. We recruited 36 respondents (undergraduates at a major university) from the lab's respondent pool via a generic, nonspecific message stating that there was a study that would take up to one hour of time. While these experiments are expensive and time consuming, they are necessary for evaluating text summarization tools and we advocate for similar studies of other methods.

5.1 Generating the sample of summaries

A choice of each of the options described above gives a unique summarizer. We evaluated 96 different summarizers built from these factors:

1. We evaluated summarizers that analyzed the data at the article-unit and paragraph-unit level (see Section 3.1).

2. When performing paragraph-unit analysis, we labeled document units using count-1, count-2, and hardcount-2. For the article-unit analysis we considered these three, plus count-3 and hardcount-3 (see Section 3.3).

3. We considered tf-idf weighting, L^2 rescaling, and simple stop-word removal (see Section 3.4).

4. We considered all four phrase-selection techniques (see Section 4).

Choices of unit of analysis, labeling, the three preprocessing options, and the four phrase-selection methods give 96 different summarizers indexed by these combinations of factors.

We compared the efficacy of these combinations by having respondents assess the quality of several different summaries generated by each summarizer. We applied each summarizer to the set of all articles in the New York Times International Section from 2009 for 15 different countries of interest. These countries are listed in Table 6. We only considered those countries with reasonable representation in the corpus (i.e., the name of the country appeared in at least 200 articles). After identifying these countries, we hand-specified a phrase set Q for each county by including any plurals and possessives of the country and any common names for the country's people. Using these 15 subjects on each of the 96 summarizers, we calculated 1,440 summaries. We consider these summaries a representative sample of the summarizers' output.

Table 6 also includes the number of positively marked examples under all the count-m labeling schemes we used for both article- and paragraph-unit analysis. The "article-1" header is the most generous labeling: any article that mentions any of the words associated with the subject one or more times is marked as treating the subject. Even under this, positive examples are scarce; it is clear we are attempting to summarize something that does not constitute a large portion of the text. Hardcount-m has the same number of positive examples as count-m, but fewer negative ones.

	article-1		article-2		article-3		paragraph-1		paragraph-2	
subject	#	%	#	%	#	%	#	%	#	%
china	1436	15%	970	10%	800	8%	6455	5%	2026	1.6%
iran	1387	15%	906	9%	715	7%	4875	4%	1621	1.2%
iraq	1139	12%	710	7%	562	6%	4806	4%	1184	0.9%
afghanistan	1133	12%	729	8%	592	6%	4774	4%	659	0.5%
israel	1126	12%	591	6%	388	4%	4478	3%	1537	1.2%
pakistan	989	10%	650	7%	555	6%	4454	3%	1384	1.1%
russia	981	10%	699	7%	590	6%	4288	3%	1168	0.9%
france	867	9%	419	4%	291	3%	2815	2%	586	0.4%
india	848	9%	613	6%	537	6%	2368	2%	559	0.4%
germany	788	8%	387	4%	284	3%	2333	2%	459	0.4%
japan	566	6%	273	3%	195	2%	1780	1%	406	0.3%
mexico	413	4%	238	2%	189	2%	1475	1%	392	0.3%
south korea	382	4%	208	2%	136	1%	1254	1%	251	0.2%
egypt	361	4%	231	2%	194	2%	1070	1%	230	0.2%
turkey	281	3%	125	1%	96	1%	797	1%	197	0.2%

Table 6: Our Experiment's Subjects With Sizes of Positive Example Sets. "#" denotes number and "%" denotes portion of units positively marked. A greater proportion of units are marked positive in the article-unit analysis. Generally, only a small portion of articles are considered topical for a given subject.

5.2 The survey and respondents

For our survey, paid respondents were convened in a large computer lab. Each respondent sat at a computer and was given a series of questions over the course of an hour. Respondents assessed a

series of summaries and articles presented in 6 blocks of 8 questions each. Each block considered a single (randomly selected) subject from our list of 15. Within a block, respondents were first asked to read four articles and rate their relevance to the specified subject. Respondents were then asked to read and rate four summaries of that subject randomly chosen from the subject's library of 96. The survey was a sequence of simple web-forms presented in a browser set to "kiosk-mode," with the navigation buttons and title hidden. Respondents could not go back to previous questions.

Before the survey all respondents did a sample series of four practice questions and were then asked if they had any questions as to how to score or rate articles and summaries.

Evaluating article topicality. To insure that respondents had a high probability of seeing several articles actually relevant to the subject being investigated, the articles presented in a block were selected with a weighted sampling scheme with weights proportional to the number of times the block's country's name was mentioned. We monitored the success of this scheme (and collected data about the quality of the automatic labelers) by asking the respondents to evaluate each shown article's relevance to the specified subject on a 1 to 7 scale.

We averaged these scores for each article to obtain overall article relevance (what we call the article's "topicality"). With 4 or higher scored as relevant, respondents saw at least 2 articles (out of the 4) on the subject of interest about 75% of the time. With 3 or higher, the number of blocks with at least two relevant articles rises to 91%. We attempt to summarize how a subject is treated overall, including how it is treated in articles in which the subject is only a secondary consideration. For example, an article focused on world energy production may still talk a bit about Russia, and connections like this are still important in driving the overall image of Russia in the news. Hence, even a modest score of 3 or 4 is a likely indication that the article has subject-related content.

An article could potentially be presented multiple times to the same respondent over the course of the survey if that article was relevant to multiple subjects. This did occur, although infrequently (only 9 articles out of 408 distinct articles presented were used for more than one subject). Only the first 120 words of each article were shown; consultation with journalists suggests this would not have a detrimental impact on content presented, as a traditional newspaper article's "inverted pyramid" structure moves from the most important information to more minute details as it progresses (Pottker, 2003).

Evaluating summaries. A simple random sample of 4 summaries were presented after the four articles. Each summary was presented on its own screen. The respondents were asked to assess each summary in four respects:

1. Content: How does this list capture the content about [subject] in the text you just read? (1-7 scale, 7 being fully captured)

2. Relevance: How many words irrelevant or unrelated to [subject] does this list contain? (1-7 scale, 7 being no irrelevance)

3. Redundancy: How many redundant or repeated words does this list contain? (1-7 scale, 7 being no redundancies)

4. Specificity: Given what you just read, would you say this list is probably too general or too specific a summary of how [subject] was covered by the newspaper in 2009? (response options: too general, about right, too specific, not relevant, and can't tell)

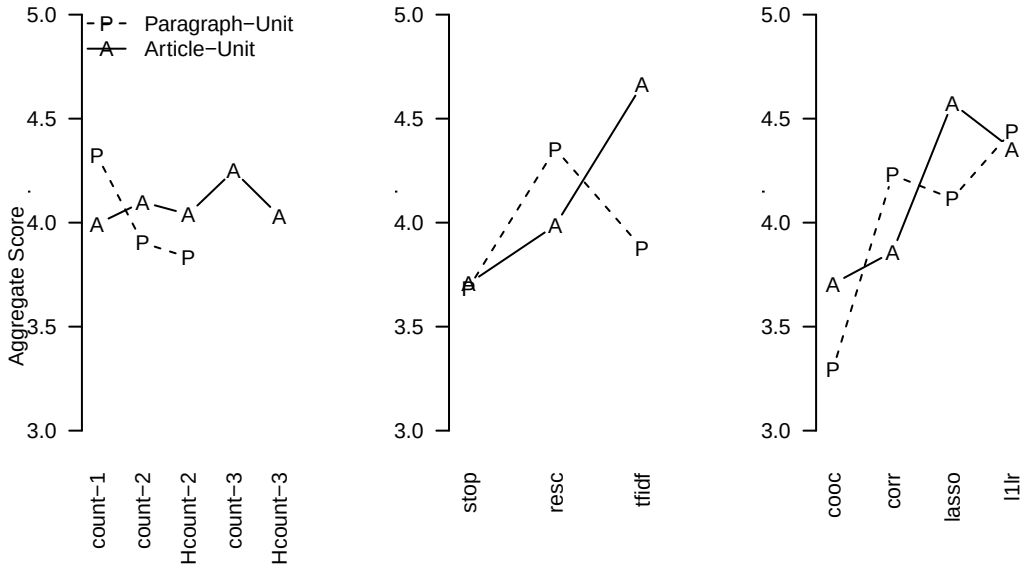

Figure 3: Comparing Article vs. Paragraph Unit Results. Interaction plots of unit of analysis vs. the other summarizer characteristics for aggregate score based on the raw data. There are major differences between article-unit analysis and paragraph-unit analysis when considering the impact of choices in preprocessing, labeling, and phrase-selection method. Note, in particular, the reversal of tf-idf and L^2 rescaling, the reversal of count-1 and count-2, and the reduction of Lasso and Co-Occur's efficacies under the paragraph-unit analysis.

All respondents finished their full survey, and fewer than 1% of the questions were skipped. Time to completion ranged from 14 to 41 minutes, with a mean completion time of 27 minutes.

6 Results

Due to significant interactions between the unit of analysis versus the other three factors (including some three-way interactions), we analyzed the article-unit and paragraph-unit data separately. See the interaction plots (made directly from the aggregate scores) for unit of analysis to the other three factors on Figure 3. We compare the characteristics of article-unit summarizers and paragraph-unit summarizers in the discussion (Section 6.4) following the individual analyses of the labeling, reweighting, and feature selection method effects for each class.

We fit the summarizer characteristics to the respondents' responses using linear regression. The models included terms for respondent, subject, unit type, rescaling used, labeling used, and feature selector used, as well as all interaction terms for the latter four features. We fit models for the three main outcomes (the Content, Relevance, and Redundancy subscores) as well as an aggregate "quality" score, taken as the mean of these three scores. The fourth question was used to examine our hypothesis that smaller document units would increase the specificity of the summaries.

There are large respondent and subject effects. Some subjects were more easily summarized than others, and some respondents were more critical than others. Interactions between the four summarizer factors are (unsurprisingly) present ($df = 33, F = 4.14, \log P \approx -13$ under ANOVA).

	Article-unit			Paragraph-unit		
Factor	Main Effect	Labeling Interact	Rescaling Interact	Main Effect	Labeling Interact	Rescaling Interact
Feature Selection	-10		-7	-6		-2
Labeling				-1		
Rescaling	-15			-3		

Table 7: Main Effects and Interactions of Factors. A number denotes a significant main effect or pairwise interaction for aggregate scores, and is the base-10 log of the P-value. Blanks denotes lack of significance.

Interaction plots suggest that the sizes of these interactions are large, making interpretation of the marginal differences for each factor potentially misleading.

6.1 Article unit analysis

Interactions between factors make interpretation difficult, but overall, Lasso is a good summarizer that is resistant to preprocessing choices. Interestingly, the simplest method, Co-Occur, is on par with Lasso under tf-idf.

The left column of Figure 4 shows plots of the three two-way interactions between feature selector, labeling scheme, and rescaling method for the article-unit data. There is a strong interaction between rescaling and phrase-selection method ($df = 6, F = 8.07, \log P \approx -8$, top-left plot), and no evidence of a labeling by phrase-selection interaction or a labeling by rescaling interaction. Model-adjusted plots (not shown) akin to Figure 4 based on the model do not differ substantially in character. Table 7 show all significant main effects and pairwise interactions. There is no significant three-way interaction.

Lasso is the most consistent method, maintaining high scores under almost all combinations of the other two factors. In Figure 4, note how Lasso has a tight cluster of means regardless of rescaling used in the first plot and how Lasso's outcomes are high and consistent across all labeling in the second plot. Though L1LR or Co-Occur may be slightly superior to Lasso when the data has been vectorized according to tf-idf, they are not greatly so, and, regardless, both these methods seem fragile, varying a great deal in their outcomes based on the text preprocessing choices. Note, for example, how vulnerable the Co-Occur phrase-selection method is to choice of rescaling.

The main effect of labeling is not statistically significant and the left plot of Figure 3 suggest the effects, if any, are not large, indicating that the choice of labeling is not critical. That said, in comparison to Lasso, the other feature selectors may be more sensitive to labeling schemes (although this interaction is also not significant under ANOVA). See the middle-left plot of Figure 4. Of interest is the apparent upward trend of labeling technique for Co-Occur; this is sensible in that more stringent labeling could potentially help block words common in less subject-relevant documents from accruing many numbers in the positive document set.

Tf-idf seems to be the best overall rescaling technique, consistently coming out ahead regardless of choice of labeling or phrase-selection method. Note how its curve is higher than the rescaling and stop-word curves in both the top- and bottom-left plots in Figure 4. Due to a correlation statistic's invariance to linear scaling, L^2-rescaling should not impact the Correlation Screening feature selector; this is confirmed by the data, with the stop-word removal (which drops some phrases, but leaves the values of the remaining columns untouched) similar to L^2-rescaling. Tf-idf

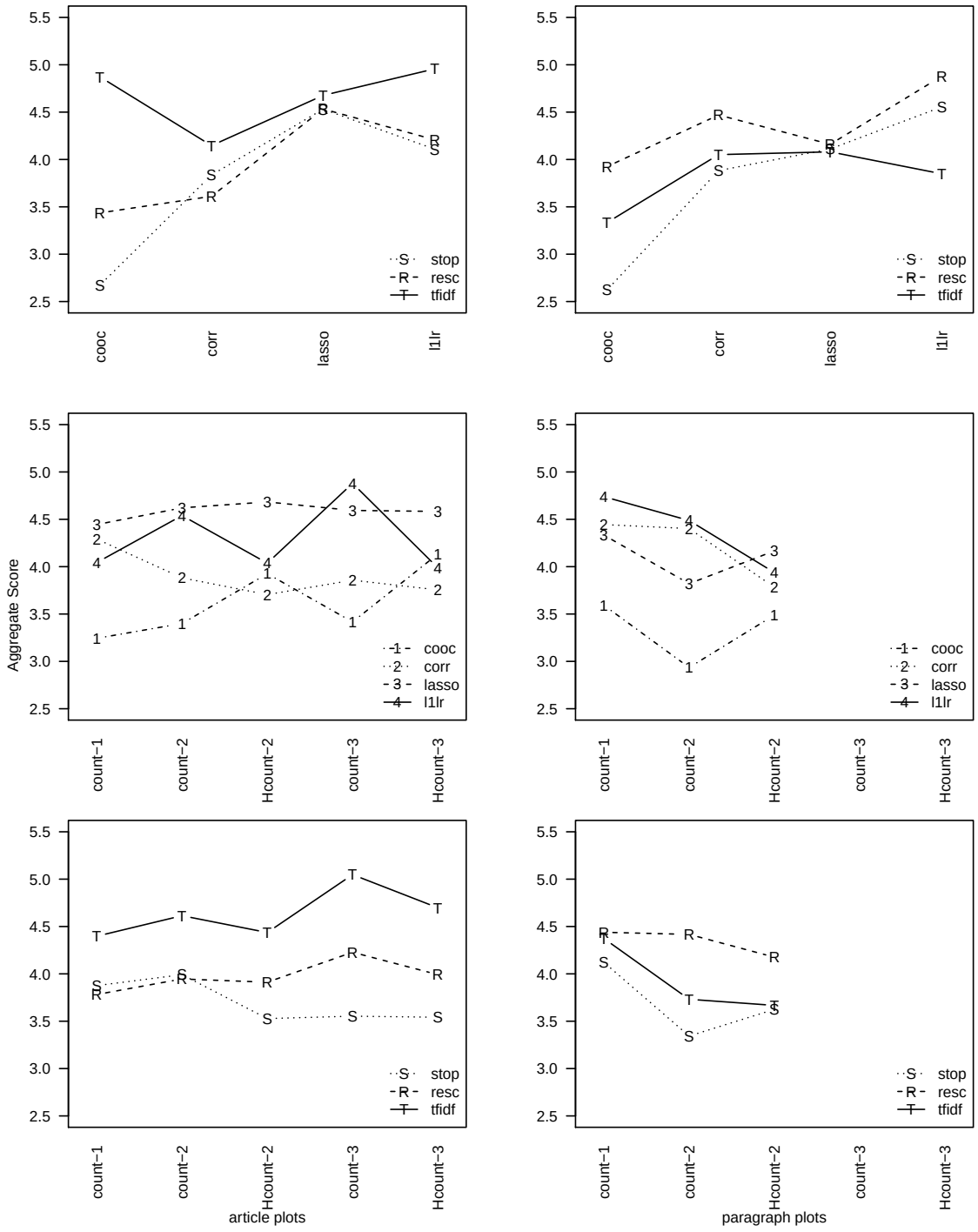

Figure 4: Aggregate Quality Plots. Pairwise interactions of feature selector, labeling, and rescaling technique. Left-hand side are for article-unit summarizers, right for paragraph-unit. See testing results for which interactions are significant.

Data Included	Order (article)	Order (paragraph)
All	cooc, corr < L1LR, Lasso stop < resc < tf-idf	cooc < corr, Lasso, L1LR tfidf, stop < resc
tf-idf only	no differences	no differences
L^2 only	cooc < L1LR, Lasso; corr < Lasso	no differences
stop only	cooc < corr, L1LR, Lasso; corr < Lasso	cooc < Lasso, L1LR
cooc only	stop < resc < tf-idf	stop < resc
corr only	stop < tf-idf	no differences
Lasso only	no differences	no differences
L1LR only	no differences	tf-idf < resc

Table 8: Quality of Feature Selectors. This table compares the significance of the separation of the feature selection methods on the margin. Order is always from lowest to highest estimated quality. A "<" denotes a significant separation. All P-values corrected for multiple pairwise testing. Note the last seven lines are lower power due to subsetting the data.

reweights by document length, and thus does introduce differences. The bottom line is that under tf-idf, all the methods seem comparable. Alternatively put, tf-idf brings otherwise poor feature selectors up to the level of the better selectors. The bottom-left plot shows that, regardless of labeling used, tf-idf does better on average than L^2 rescaling, which in turn does as well or slightly better than stop-word removal.

Adjusting P-values with Tukey's honest significant difference and calculating all pairwise contrasts for each of the three factors show which choices are overall good performers, ignoring interactions. For each factor, we fit a model with no interaction terms for the factor of interest and then performed pairwise testing, adjusting the P-values to control familywise error rate. See Table 8 for the resulting rankings of the factor levels. Co-Occur and Correlation Screening are significantly worse than L1LR and Lasso (correlation vs. L1LR gives $t = 3.46, P < 0.005$). The labeling method options are indistinguishable. The rescaling method options are ordered with tf-idf significantly better than rescaling ($t = 5.08, \log P \approx -4$), which in turn is better than stop-word removal ($t = 2.45, P < 0.05$).

6.2 Paragraph unit analysis

For the paragraph-unit summarizers, the story is similar. Lasso is again the most stable to various pre-processing decisions, but does not have as strong a showing under some of the labeling choices. Co-Occur is again the most unstable. L1LR and Correlation Screening outperform Lasso under some configurations. The main difference from the article-unit data is that tf-idf is a poor choice and L^2-rescaling is the best choice. Stop-word removal remains an inferior choice.

The right column of Figure 4 shows the interactions between the three factors. There is again a significant interaction between rescaling and method ($df = 6, F = 3.25, P < 0.005$, top-plot). This time, however, it is not entirely due to Co-Occur being sensitive to rescaling. Co-Occur is still sensitive, but correlation and L1LR are as well. Stop-word removal does quite well for L1LR and Lasso, suggesting that rescaling is less relevant for shorter text units.

Co-Occur is significantly worse than the other three on the margin (Co-Occur vs. Correlation Screening gives an adjusted pairwise test with $t = 4.11, P < 0.0005$), but the other three are indistinguishable. Labeling matters significantly ($df = 2, F = 5.23, P < 0.01$), with count-1 doing

better in the margin than count-2 and hardcount-2. The higher threshold is likely removing too many substantive paragraphs from the set of positive examples. See Table 6—around 75% of the examples are dropped by moving from count-1 to count-2.

6.3 Analysis of subscores

The above analysis considers the aggregate score across (1) specific *Content* captured, (2) *Redundancy* of phrases in the list, and (3) general *Relevance* of phrases in the list to the subject. We also performed the above analyses for each of the three sub-scores separately. Overall conclusions mainly hold, with a few important exceptions. The article-unit results across the three subscores are more consistent, but the paragraph-unit results are not, suggesting that paragraph-unit analysis requires more fine-tuning to get good results.

The Lasso and L1LR maintain word-lists that have few repeats, but their information capture degrades when given quite short units of text. This partially explains the weaker performance of Lasso in the aggregate scores for the paragraph-unit. For the paragraph unit summarizers, L^2-Rescaling is clearly superior for Relevance and Content scores, but inferior to tf-idf for Redundancy.

Content. The Content scores in the article-unit data are not as differentiated by feature selection method as compared to the Aggregate scores, although the trends are consistent. In the paragraph-unit data, Lasso's scores are not high but L1LR remains strong. The pattern of interactions under Aggregate scores and under Content scores is the same.

Redundancy. The marginal Redundancy scores for feature selection method are extremely differentiated, with L1LR and Lasso both scoring high and Co-occur and Correlation Screening scoring quite low. Under the article-unit data, L1LR is extremely high, suggesting that few, if any, redundant phrases are selected. Correlation Screening, on the other hand, does terribly. This is probably because partially overlapping phrases tend to have similar correlations with the labeling y, and thus Correlation Screening picks clusters of them up, even with the pruning of sub-phrases. See, for example, "prime minister wen jiabao" on Table 3. Interestingly, in the paragraph-unit analysis, tf-idf is the superior rescaling choice for Redundancy, even though it is not for the Aggregate scores.

Relevance. The trends in the Relevance scores are the same as for the Aggregate scores. There is slightly greater separation between L^2-rescaling, tf-idf reweighting, and stop-word removal in the paragraph unit analysis. Co-occur is an even worse performer under this metric as compared to Aggregate.

Correlation Screening's poor Redundancy score substantially reduces its aggregate score. This might be solved by a different, more sophisticated, pruning technique. Indeed, given Correlation Screening's quite high scores for Relevance and its strong showing for the paragraph-unit data for both Relevance and Content, fixing the Redundancy problem could result in a good, fast summarizer that may well outperform the penalized regression methods.

6.4 Discussion

The feature selectors interact differently with labeling and rescaling under the two different units of analyses. While the overall summary quality was no different between these two varieties of summarizer, interaction plots suggest labeling is important, with count-2 being more appropriate

for articles and count-1 being more appropriate for paragraph units (see top plots of Figure 3). This is unsurprising: a count of 1 vs. 2 means a lot more in a single paragraph than an entire article.

Preprocessing choice is a real concern. While stop-word removal and L^2-rescaling seem relatively consistent across both units of analysis, tf-idf works much worse, overall, for the paragraph unit summarizers than with articles. This is probably due to the short length of the paragraph causing rescaling by term frequency to have large and varying impact. It might also have to do with tf-idf correctly adjusting for the length of the short "World-Briefing" articles. Under Lasso, however, these decisions seem less important, regardless of unit size.

Comparing the performance of the feature selectors is difficult due to the different nature of interactions for paragraph and article units. That said, Lasso consistently performed well. For the article-unit it performed near the top. For the paragraph-unit it did better than most but was not as definitively superior. L1LR, if appropriately staged, also performs well, although it has a much higher computational cost.

We hypothesized that paragraph-unit analysis would generate more specific summaries and article-unit more general. This does not seem to be the case; in analyzing the results for the fourth question on generality vs. specificity of the summaries (not shown), there was no major difference found between article-unit and paragraph-unit summarizers.

It is on the surface surprising that the Lasso often outperformed L1LR as L1LR fits a model that is more appropriate for the binary outcome of the labeling. The Lasso has a L^2-loss, which is sensitive to outliers, while L1LR's logistic curve is less sensitive. However, the design matrix X, especially under rescaling, is heavily restricted. All entries are nonnegative and few are large. This may limit the opportunity for individual entries in the L^2 loss to have a significant impact, ameliorating the major drawback of the Lasso.

There is no evidence that dropping units that mention the subject below a given threshold (the hardcount labeling technique) is a good idea. Indeed, it appears to be a bad one. The pattern of a quality dip between count-n and hardcount-n appears both in the paragraph- and article-unit results. Perhaps articles that mention a subject only once are important negative examples. The sub-scores offer no further clarity on this point.

7 Conclusions

News media significantly impacts our day to day lives and the direction of public policy. Analyzing the news, however, is a complicated task. The labor intensity of hand coding either leads to small-scale studies, or great expense. This and the amount of news available to a typical consumer strongly motivate automated methods.

We proposed a sparse predictive framework for extracting meaningful summaries of specific subjects from a media corpus. Under this framework, depending on the preprocessing scheme, document-granularity, labeling choice, and feature selection methods, several summarizers are proposed and then evaluated with a human validation experiment.

Based on the human experiment, we conclude that the features selected using a prediction framework do generally form an informative key-phrase summary of a subject of interest. In particular, the vector representation of the data should incorporate some reweighting of the phrase appearance counts. Tf-idf is a good overall choice unless the text units are small (e.g., paragraphs, and, presumably, headlines, online comments, and tweets) when an L^2 scaling should be used.

The Lasso is a good overall feature selector that seems robust to how the data is vectorized and labeled and it is computationally scalable. L1LR, a natural fit model-wise, can perform well if preprocessing is done correctly. However, it is computational expensive.

Limiting phrases to three words or fewer is a potential problem; we encountered it, for example, when dealing with political leaders frequently mentioned with title (as in "Secretary of State Hillary Clinton"). Ifrim et al. (2008) proposed an algorithm for L1LR that allows for arbitrary-length key-phrases, but it does not allow for intercept or reweighting. Alternatively, natural language tools such as parts of speech tagging could pull out such names as distinct features. This alternate approach is currently under investigation.

Our frameowrk provides a general tool for text summarization. We are now in the process of working with social scientists to use this tool to carry out research in a substantive social science field.

8 Acknowledgements

We are extremely grateful to Hoxie Ackerman and Saheli Datta for help assembling this publication. We are indebted to the staff of the XLab at UC Berkeley for their help in planning and conducting the human validation study. This work was partially supported by the National Science Foundation grant SES-0835531 under the "Cyber-Infrastructure and Discovery program." Luke Miratrix is grateful for the support of a Graduate Research Fellowship from the National Science Foundation.

References

Blei, D. and J. McAuliffe (2008). Supervised topic models. In J. Platt, D. Koller, Y. Singer, and S. Roweis (Eds.), *Advances in Neural Information Processing Systems 20*, pp. 121–128. Cambridge, MA: MIT Press.

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). Latent Dirichlet allocation. *J. Mach. Learn. Res. 3*, 993–1022.

Branton, R. P. and J. Dunaway (2009). Slanted newspaper coverage of immigration: The importance of economics and geography. *Policy Studies Journal 37*(2), 257–273.

Chang, J., J. Boyd-Graber, S. Gerrish, C. Wang, and D. Blei (2009). Reading tea leaves: How humans interpret topic models. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta (Eds.), *Advances in Neural Information Processing Systems 22*, pp. 288–296.

Chen, J., Q. Yang, B. Zhang, Z. Chen, D. Shen, and Q. Cheng (2006). Diverse topic phrase extraction from text collection. In *WWW 2006*, Edinburgh, UK.

Denham, B. E. (2004). Hero or hypocrite? *International Review for the Sociology of Sport 39*(2), 167–185.

El Ghaoui, L., V. Viallon, and T. Rabbani (2011). Safe feature elimination for the LASSO. *Journal of Machine Learning Research*. Submitted, April 2011.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res. 3*, 1289–1305.

Frank, E., G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning (1999). Domain-specific keyphrase extraction. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, California, pp. 668–673. Morgan Kaufmann.

Gawalt, B., J. Jia, L. W. Miratrix, L. Ghaoui, B. Yu, and S. Clavier (2010). Discovering word associations in news media via feature selection and sparse classication. In *MIR '10*, Proceedings of the International Conference on Multimedia Information Retrieval, Philadelphia, Pennsylvania, USA, pp. 211–220.

Genkin, A., D. D. Lewis, and D. Madigan (2007). Large-scale bayesian logistic regression for text categorization. *Technometrics 49*(3), 291–304.

Gilens, M. and C. Hertzman (2000). Corporate ownership and news bias: Newspaper coverage of the 1996 telecommunications act. *The Journal of Politics 62*(02), 369–386.

Goldstein, J., V. Mittal, J. Carbonell, and M. Kantrowitz (2000). Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop on Automatic summarization*, pp. 40–48.

Hastie, T., R. Tibshirani, and J. H. Friedman (2003). *The Elements of Statistical Learning*. unknown: Springer.

Hennig, L. (2009). Topic-based multi-document summarization with probabilistic latent semantic analysis. In *Recent Advances in Natural Language Processing (RANLP)*.

Ifrim, G., G. Bakir, and G. Weikum (2008). Fast logistic regression for text categorization with variable-length n-grams. In *14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, pp. 354–362. ACM.

Lazer, D., A. Pentland, L. Adamic, S. Aral, A.-L. Barabsi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. Van Alstyne (2009). Computational social science. *Science 323*(5915), 721–723.

Lee, L. and S. Chen (2006). New methods for text categorization based on a new feature selection method and a new similarity measure between documents. *Lecture Notes in Computer Science 4031*, 1280.

Miller, J. M. and J. A. Krosnick (2000). News media impact on the ingredients of presidential evaluations: Politically knowledgeable citizens are guided by a trusted source. *American Journal of Political Science 44*(2), pp. 301–315.

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

Neto, J., A. Freitas, and C. Kaestner (2002). Automatic text summarization using a machine learning approach. *Advances in Artificial Intelligence 2507*, 205–215.

Nisbet, E. C. and T. A. Myers (2010). Challenging the state: Transnational TV and political identity in the middle east. *Political Communication 27*(4), 347–366.

Potter, A. E. (2009). Voodoo, zombies, and mermaids: U.S. newspaper coverage of Haiti. *Geographical Review 99*(2), 208 – 230.

Pottker, H. (2003). News and its communicative quality: the inverted pyramid–when and why did it appear? *Journalism Studies 4*, 501–511(11).

Rose, S., D. Engel, N. Cramer, and W. Cowley (2010). Automatic keyword extraction from individual documents. In M. W. Berry and J. Kogan (Eds.), *Text Mining: Applications and Theory*. unknown: John Wiley and Sons, Ltd.

Salton, G. (1991). Developments in automatic text retrieval. *Science 253*(5023), 974–980.

Senellart, P. and V. D. Blondel (2008). Automatic discovery of similar words. In *Survey of Text Mining II*. unknown: Springer.

Wahl, O. E., A. Wood, and R. Richards (2002). Newspaper coverage of mental illness: Is it changing? *Psychiatric Rehabilitation Skills 6(1)*, 9–31.

Yang, Y. and I. O. Pendersen (1997). A comparative study on feature selection in text categorization. In *ICML-97, 14th International Conference on Machine Learning*, Nashville, US, pp. 412–420.

Zhang, T. and F. J. Oles (2001). Text categorization based on regularized linear classfiication methods. *Information Retrieval 4*, 5–31.