Understanding Large Text Corpora via Sparse Machine Learning

Laurent El Ghaoui* Vu Pham† Guan-Cheng Li‡ Viet-An Duong§
Ashok Srivastava¶ Kanishka Bhaduri‖

November 30, 2012

Abstract

Sparse machine learning has recently emerged as powerful tool to obtain models of high-dimensional data with high degree of interpretability, at low computational cost. The approach has been successfully used in many areas, such as signal and image processing. This paper posits that these methods can be extremely useful in the analysis of large collections of text documents, without requiring user expertise in machine learning. Our approach relies on three main ingredients: (a) multi-document text summarization and (b) comparative summarization of two corpora, both using sparse regression or classification; (c) sparse principal components and sparse graphical models for unsupervised analysis and visualization of large text corpora. We validate our methods using a corpus of Aviation Safety Reporting System (ASRS) reports and demonstrate that the methods can reveal causal and contributing factors in runway incursions. Furthermore, we show that the methods automatically discover four main tasks that pilots perform during flight, which can aid in further understanding the causal and contributing factors to runway incursions and other drivers for aviation safety incidents. We also provide a comparative study involving other commonly used datasets, and report on the competitiveness of sparse machine learning compared to state-of-the-art methods such as Latent Dirichlet Allocation (LDA).

*EECS Dept., UC Berkeley, elghaoui@berkeley.edu.
†EECS Dept., UC Berkeley, ptvu@berkeley.edu.
‡EECS Dept., UC Berkeley, guanchengli@berkeley.edu.
§Ecole des Mines d'Alès, School of Production & Systems Engineering, viet-an.duong@mines-ales.org.
¶System-Wide Safety and Assurance Technologies Project, NASA, ashok.n.srivastava@nasa.gov.
‖System-Wide Safety and Assurance Technologies Project, NASA, kanishka.bhaduri-1@nasa.gov.

Contents

1 Introduction

Sparse machine learning refers to a set of learning algorithms that seek a trade-off between some goodness-of-fit measure and sparsity of the result, the latter property allowing better interpretability. In a sparse learning classification task for example, the prediction accuracy or some other classical measure of performance is not the sole concern: we also wish to be able to better understand which few features are relevant as markers for classification. Thus, if a binary classification task involves, say, data with genes as features, one wishes to provide not only a high-performance classifier, but one that only involves a few genes, allowing biologists to focus their further research efforts on those specific genes. Binary classification algorithms often provide a weight for each feature, hence if the weight vector is sparse (it contains many zero weights), then the features with non-zero weights are the ones that are the most relevant in understanding the difference between the two classes. Similar benefits are derived from sparsity in the context of unsupervised learning, as discussed in more detail later.

There is an extensive literature on the topic of sparse machine learning, with terms such as compressed sensing, l_1-norm penalties and convex optimization [15, 9, 4, 8, 54, 8, 47], often associated with the topic. Successful applications of sparse methods have been reported, mostly in image and signal processing, see for example [19, 33, 35]. Due to the intensity of research in this area, many very efficient algorithms have been developed for sparse machine learning in the recent past. Despite an initial agreement that sparse learning problems are more computationally difficult than their non-sparse counterparts, a new consensus might soon emerge that sparsity constraints or penalties actually *help* reduce the computational burden involved in learning.

Our paper makes the claim that sparse learning methods can be very useful to the understanding of large *text* databases. Of course, machine learning methods in general have already been successfully applied to text classification and clustering, as evidenced by a large body of literature, for example by [26]. We show that sparsity is an important added property that is a crucial component in any tool aiming at providing *interpretable* statistical analysis, allowing in particular efficient multi-document summarization, comparison, and visualization of huge-scale text corpora. More classical algorithms, such as naïve Bayes (for supervised learning tasks) and Latent Dirichlet Association (for unsupervised learning, see [7]), can also be applied for such tasks. However, these algorithms do not incorporate sparsity directly into the model, and applying them for the text processing tasks considered here requires a final "thresholding" step to make the result interpretable. The experiments in this paper indicate that the sparse learning approaches provide an efficient alternative to these popular models, and in the case of LDA, at a fraction of the computational cost, and much better readability of the code.

To illustrate our approach, we perform an analysis, using our methods, of a specific data set coming from the Aviation Safety Reporting System (ASRS) database. This database contains reports generated by pilots, air traffic controllers, and others on a voluntary basis, and is a crucial component of the continuing effort to maintain and improve aviation safety[1]. The ASRS data contains several of the crucial and generic challenges involved under the general banner of "large-scale text data understanding". First, its scale is huge, and growing rapidly, making the need for automated analyses of the processed reports more crucial than ever. Another issue is that the reports themselves are far from being syntactically correct, with lots of abbreviations, orthographic and grammatical errors, and other shortcuts. Thus we are not facing a corpora with well-structured language having clearly defined rules, as we would if we were to consider a corpus of laws or bills or any other well-redacted data set. Finally, in many cases we do not know in advance what to look for, because the goal is to discover precursors to aviation safety incidents and accidents. In other words, the task is not about search, and finding a needle in a haystack: in many cases, we cannot simply monitor the emergence or disappearance of a few keywords that would be known in advance. Instead the task resembles more one of trying to visualize the haystack itself, compare various parts of it, or summarize some areas.

Our main goal in this paper is to illustrate how we can use the different algorithms in the sparse machine learning toolbox, in order to gain a better understanding of a data set such as the ASRS corpora. An

[1] See `http://asrs.arc.nasa.gov` for more information on the ASRS system. The text reports are available on that website.

additional goal is to provide a comparative study of LDA and a sparse unsupervised learning method called sparse PCA.

Our paper, which is an extended version of the conference paper [17], is organized as follows. Section 2 is devoted to a review of some of the main models and algorithms in sparse machine learning. We also explain how to apply the sparse learning models to important text processing tasks such as topic summarization. Section 3 illustrates the sparse learning approach in the context of ASRS data, and also reviews some prior work on this specific data set. Section 4 provides a comparative study of sparse PCA and LDA, on several popular text data sets.

2 Sparse Learning Methods

In this section we review some of the main algorithms of sparse machine learning, and then explain how these models can be used for some generic tasks arising in text analysis.

2.1 Sparse classification and regression

LASSO Regression. Perhaps the most well known example of sparse learning is the variant of least-squares known as the LASSO [46], which takes the form

$$\min_{\beta} \ \|X^T\beta - y\|_2^2 + \lambda\|\beta\|_1, \tag{1}$$

where X is a $n \times m$ data matrix (with each row a specific feature, each column a specific data point), y is a m-dimensional response vector, and $\lambda > 0$ is a parameter. The l_1-norm penalty encourages the regression coefficient vector β to be sparse, bringing interpretability to the result. Indeed, if each row is a feature, then a zero element in β at the optimum of (1) implies that that particular feature is absent from the optimal model. If λ is large, then the optimal β is very sparse, and the LASSO model then allows to select the few features that are the best predictors of the response vector.

The LASSO problem looks more complicated than its classical least-squares counterpart. However, there is mounting evidence that, contrary to intuition, the LASSO is substantially *easier* to solve than least-squares, at least for high values of the penalty parameter λ. As shown later, in typical applications to text classification, a high value of λ is desired, which is precisely the regime where the LASSO is computationally very easy to solve. The so-called safe feature elimination procedure, introduced in [18], allows to cheaply detect that some of the components of β will be zero at optimum. This in turn enables to treat data sets having millions of terms and documents, at least for high values of λ.

Many algorithms have been proposed for LASSO; at present it appears that, in text applications with sparse input matrix X, a simple method based on minimizing the objective function of (1) one coordinate of β at a time is extremely competitive [21, 37].

Other loss functions. Similar models arise in the context of support vector machines (SVM) for binary classification, where the sparse version takes the form (see e.g. [5])

$$\min_{\beta,b} \ \frac{1}{m}\sum_{i=1}^{m} h(y_i(x_i^T\beta + b)) + \lambda\|\beta\|_1, \tag{2}$$

where now y is the vector of ± 1's indicating appartenance to one of the classes, and h is the so-called hinge loss function, with values $h(t) = \max(0, 1-t)$. At optimum of problem (2), the above model parameters (β, b) yield a classification rule, *i.e.* predict a label $\hat{y}$ for a new data point x, as follows: $\hat{y} = \mathbf{sign}(x^T\beta + b)$. A smooth version of the above is sparse logistic regression, which obtains upon replacing the hinge loss with a smooth version $l(t) = \log(1 + e^{-t})$. Both of these models are useful but somewhat less popular than the LASSO, as state-of-the-art algorithms are have not yet completely caught up. For our text applications, we have found that LASSO regression, although less adapted to the binary nature of the problem, is still very efficient [23].

2.2 Sparse principal component analysis

The classical Principal Component Analysis (PCA) method allows to reduce the dimension of data sets by performing a low-rank approximation to the data matrix, and projecting data points on the corresponding subspace. Sparse principal component analysis (Sparse PCA, see [55, 52] and references therein) is a variant of PCA that allows to find sparse directions of high variance. The sparse PCA problem can be formulated in many different ways, one of them (see [43, 32]) involves a low-rank approximation problem where the sparsity of the low-rank approximation is penalized:

$$\min_{p,q} \|M - pq^T\|_F^2 \ : \ \|p\|_0 \leq k, \ \ \|q\|_0 \leq h, \tag{3}$$

where M is the $m \times n$ data matrix, $\|\cdot\|_F$ is the Frobenius norm. In the above, the notation $\|\cdot\|_0$ stands for the cardinality, that is, the number of non-zeros in its vector argument, and $k \leq m$, $h \leq n$ are parameters that constrain the cardinality of the solution (p, q). Classical PCA is obtained with $k = m, h = n$.

The model above results in a rank-one approximation to M (the matrix pq^T at optimum), and vectors p, q are constrained to be sparse. If M is a term-by-document matrix, the above model provides sparsity in the feature space (via p) and the document space (via a "topic model" q), allowing to pinpoint a few features and a few documents that jointly "explain" data variance.

Several algorithms have been proposed for the above problem, or related variants, see for example [28, 43, 11]. The approach in [53] is based on solving a relaxation to the problem, one column of the matrix variable at a time. Other algorithms (*e.g.* [43]) attempt to solve the problem directly, without any relaxation; these kinds of methods are not guaranteed to even converge to a local minimum. However, they appear to be quite efficient in practice, and extremely scalable. One such algorithm consists in solving the above problem alternatively over p, q many times [43]. This leads to a modified power iteration method

$$p \to P(T_k(Mq)), \ \ q \to P(T_h(M^T p)),$$

where P is the projection on the unit circle (assigning to a non-zero vector v its scaled version $v/\|v\|_2$), and for $t \geq 0$, T_t is the "hard thresholding" operator (for a given vector v, $T_t(v)$ is obtained by zeroing out all but the t largest components of v).

In some applications, involving for example visualization of large text databases, it is useful to distinguish positive and negative components of vectors p, q, and retain the a fixed number of the largest positive and largest negative components separately. We further elaborate on this point in Section 2.5, and illustrate this in Section 3.

With $k = m, h = n$, the original power iteration method for the computation of the largest singular value of M is recovered, with optimal p, q the right- and left- singular vectors of M. The presence of cardinality constraints modifies these singular vectors to make them sparser, while maintaining the closeness of M to its rank-one approximation. The hard-thresholding version of power iteration scales extremely well with problem size, with greatest speed increases over standard power iteration for PCA when a high degree of sparsity is asked for. This is because the vectors p, q are maintained to be extremely sparse during the iterations.

An alternative algorithm for solving the above is based on solving a classical PCA problem, then thresholding the resulting singular vectors so that they have the desired level of sparsity. (We discuss "thresholded models" in more details in Section 2.4.) For large-scale data, PCA is typically solved with power iteration, so the "thresholded PCA" algorithm is very similar to the above thresholded power iteration for sparse PCA. The only difference is in how many times thresholding takes place. Note that in practice, the thresholded power iteration for sparse PCA is much faster than its plain counterpart, since we are dealing with much sparser vectors as we perform the power iterations.

2.3 Sparse graphical models

Sparse graphical modeling seeks to uncover a graphical probabilistic model for multivariate data that exhibits some sparsity characteristics. One of the main examples of this approach is the so-called sparse covariance

selection problem, with a Gaussian assumption on the data (see [38], and related works such as [22, 34, 50, 44, 31, 29]). Here we start with a $n \times n$ sample covariance matrix S, and assuming the data is Gaussian, formulate a variant to the corresponding maximum likelihood problem:

$$\max_X \log\det X - \mathbf{Tr}SX - \lambda\|X\|_1, \tag{4}$$

where $\lambda > 0$ is a parameter, and $\|X\|_1$ denotes the sum of the absolute values of all the entries in the $n \times n$ matrix variable X. Here, $\mathbf{Tr}SX$ is the scalar product between the two symmetric matrices S and X, that is, the sum of the diagonal entries in the matrix product SX. When $\lambda = 0$, and assuming S is positive-definite, the solution is $X = S^{-1}$. When $\lambda > 0$, the solution X is always invertible (even if S is not), and tends to have many zero elements in it as λ grows. A zero element in the (i, j) entry of X corresponds to the conditional independence property between nodes i and j; hence sparsity of X is directly related to that of the conditional independence graph, where the absence of an edge denotes conditional independence.

The covariance selection problem is much more challenging than its classical counterpart (where $\lambda = 0$), which simply entails inverting the sample covariance matrix. At this point it appears that one of the most competitive algorithms involves solving the above problem one column (and row) of X at a time. Each sub-problem can be interpreted as a LASSO regression problem between one particular random variable and all the others [38, 22]. Successful applications of this approach include Senate voting [38] and gene data analysis [38, 14]

Just as in the PCA case, there is a conceptually simple algorithm, which relies on thresholding. If the covariance matrix is invertible, we simply invert it and threshold the elements of the inverse. Some limited evidence points to the statistical superiority of the sparse approach (based on solving problem (4)) over its thresholded counterpart.

2.4 Thresholded models

The algorithms in sparse learning are built around the philosophy that sparsity should be *part of the model's formulation*, and not produced as an afterthought. Sparse modeling is based on some kind of direct formulation of the original optimization problem, involving, typically, an l_1-norm penalty. As a result of the added penalty, sparse models have been originally thought to be substantially more computationally challenging than their non-penalized counterparts.

In practice, sparse results can be obtained after the use of almost any learning algorithm, even one that is not necessarily sparsity-inducing. Sparsity is then simply obtained via thresholding the result. This is the case for example with naïve Bayes classification: since a naïve Bayes classifier assigns weights to each feature, we can simply zero out the smaller weights to obtain a sparse classification rule. The same is true for unsupervised algorithms such as Latent Dirichlet Allocation (LDA, see [6]). In the case of LDA, the result is a probability distribution on all the terms in the dictionary. Only the terms with the highest weights are retained, which amounts in effect to threshold the probability distribution. The notion of *thresholded models* refers to the approach of applying a learning algorithm and obtaining sparsity with a final step of thresholding.

The question about which approach, "direct" sparse modeling or sparse modeling via thresholding, works better in practice, is a natural one. Since direct sparse modeling appears to be more computationally challenging, why bother? Extensive research in the least-squares case shows that thresholding is actually often sub-optimal [23]. Similar evidence has been reported on the PCA case [52]. Our own experiments in section 3 support this viewpoint.

There is an added benefit to direct sparse modeling—a computational one. Originally thresholding was considered as a computational shortcut, a quick way sparse models. As we argued above for least-squares, SVM and logistic regression, and PCA, sparse models can be actually surprisingly easier to solve than classical models; at least in those cases, there is no fundamental reason for insisting on thresholded models, although they can produce good results. For the case of covariance selection, the situation is still unclear, since direct sparse modeling via problem (4) is still computationally challenging.

The above motivates many researchers to "sparsify" existing statistical modeling methodologies, such as LDA. Note that LDA also encodes a notion of sparsity, not in the feature space, but on the document (data) space: it assumes that each document is a mixture of a small number of topics, where the topic distribution is assumed to have a Dirichlet prior. Thus, depending on the concentration parameter of this prior, a document comprised of a given set of words may be effectively restricted to having a small number of topics.

This notion of sparsity (document-space sparsity) does not constrain the number of features active in the model, and does not limit overall model complexity. As a result, in LDA, the inclusion of terms that have little discrimination power between topics (such as 'and', 'the', etc) may fall into multiple topics unless they are eliminated by hand. Once a set of topics is identified the most descriptive words are depicted as a list in order of highest posterior probability given the topic. As with any learning method, thresholding can be applied to this list to reveal the top most descriptive words given a topic. It may be possible to eliminate this thresholding step using a modified objective function with an appropriate sparsity constraint. This is an area of very active research, as evidenced by [16].

2.5 Applying sparse machine learning to text

In this section, we review some of the text processing tasks that can be addressed using sparse learning methods.

Topic summarization. Topic summarization is an extensive area of research in natural language processing and text understanding. For a recent survey on the topic, see [10]. There are many instances of this problem, depending on the precise task that is addressed. For example the focus could be to summarize a single unit of text, or summarize multiple documents, or summarize two classes of documents in order to produce the summaries that offer the best contrast. Some further references to summarization include [24, 25, 36].

The approach introduced in [23] relies on LASSO regression to produce a summary of a particular topic as treated in multiple documents. This is part of the *extraction* task within a summarization process, where relevant terms are produced and given verbatim [10]. Using predictive models for topic summarization has a long history, see for example [41]; the innovation is the systematic reliance on *sparse* regression models.

The basic idea is to divide the corpora in two classes, one that corresponds to the topic, and the other to the rest of the text corpora. For example, to provide the summary of the topic "China" in a corpora of news articles from *The New York Times* over a specific period, we may separate all the paragraphs that mention the term "china" (or related terms such as "chinese", "china's", etc) from the rest of the paragraphs. We then form a numerical, matrix representation X (via, say, TF-IDF scores) of the data, and form a "response" vector (with 1's if the document mentions China and -1 otherwise). Solving the LASSO problem (1) leads to a vector β of regressor coefficients, one for each term of the dictionary. Since LASSO encourages sparsity, many elements of β are zero. The non-zero elements point to terms in the dictionary that are highly predictive of the appearance of "china" in any paragraph in the corpus.

The approach can be used to contrast two sets of documents. For example, we can use it to highlight the terms that allow to best distinguish between two authors, or two news sources on the same topic.

Topic summarization is closely related to *topic modeling* via Latent Dirichlet Allocation (LDA) [6], which finds on a latent probabilistic model to produce a probability distribution of all the words. Once the probability distribution is obtained, the few terms that have the highest probability are retained, to produce some kind of summary in an unsupervised fashion. As discussed in section 2.4, the overall approach can be seen as a form of indirect, thresholding method for sparse modeling.

Discrimination between several corpora. Here the basic task is to find out what terms best describe the differences between two or more corpora. We simply classify one of the corpora against all the others: the (say) positive class will contain all the documents from one corpora, and the negative class includes the documents from all the remaining corpora. We can use any sparse binary classification algorithm for the

task, included the thresholded models referred to in section 2.4. The classification algorithm will identify the features that are most relevant in distinguishing a document from one class (the corpora under study) to one from the other class.

The resulting classifier weight vector, which is sparse, then points to a short list of terms that are most representative of the salient differences between the corpora and all the others. Of course, related methods such as multi-class sparse logistic regression can be used.

Visualization and clustering. Sparse PCA and sparse graphical models can provide insights to large text databases. PCA itself is a widely used tool for data visualization, but as noted by many researchers, the lack of interpretability of the principal components is a challenge. A famous example of this difficulty involves the analysis of Senate voting patterns. It is well-known in political science that, in that type of data, the first two principal components explain the total variance very accurately [38]. The first component simply represents party affiliation, and accounts for a high proportion of the total variance (typically, 80%). The second component is much less interpretable.

Using sparse PCA, we can provide axes that are sparse. Concretely this means that they involve only a few features in the data. Sparse PCA thus brings an interpretation, which is given in terms of which few features explain most of the variance. As mentioned before, it is possible to assign a fixed number of terms to each axis direction, one for the positive and one for the negative directions. (We illustrate this in our experiments on the ASRS data set.) Likewise, sparse graphical modeling can be very revealing for text data. Because it produces sparse graphs, it can bring an understanding as to which variables (say, terms, or sources, or authors) are related to each other and how.

3 Experimental Analysis on ASRS Data

3.1 Goals of the study

In this section our focus is on reports from the Aviation Safety Reporting System (ASRS). The ASRS is a voluntary program in which pilots, co-pilots, other members of the flight crew, flight controllers, and others file a text report to describe any incident that they may have observed that has a bearing on aviation safety. Because the program is completely voluntary and the data are de-identified, meaning that the author, his or her position, the carrier, and other identifying information is not available in the report. After reports are submitted, analysts from ASRS may contact the author to obtain clarifications. However, the information provided by the reporter is not investigated further. This motivates the use of (semi-) automated methods for the real-time analysis of the ASRS data. In our experiments, we have used the one provided by NASA as part of the SIAM 2007 Text Mining Competition. It consists in about 20,000 flight reports submitted by pilots after their flight. Each report is a small paragraph describing any incident that was recorded during flight, and is assigned a category (totaling 22), or type of incident.

Our goals here are as follows. A first objective is to report on previous work on this particular data set (Section 3.2). Then in Section 3.3, our aim is to validate our methods based on categorical information. Using our comparative summarization methods, we investigate if we can recover summaries for each category that allow to clearly distinguish between them, and are consistent with their meaning.

In Section **??**, we illustrate how sparse PCA can be used to visualize the data, specifically visualize the different categories. We also make a comparison with threshdolded LDA.

In Section 3.4, we focus on the analysis of runway incursions, which are events in which one aircraft moves into the path of another aircraft during landing or takeoff. A key question that arises in the study of runway incursions is to understand whether there are significant distinguishing features of runway incursions for different airports. Although runway incursions are common, the causes may differ with each airport. These are the causal factors that enable the design of the intervention appropriate for that airport, whether it may be runway design, runway lighting, procedures, etc. To do this kind of analysis, we further processed the ASRS data a bit more, as detailed in Appendix A.

3.2 Related work on ASRS data

In this section we list some previous work in applying data mining/machine learning methods for analyzing ASRS data, along with pointers for further research.

Text Cube [30] and Topic Cube [51] are multi-dimensional data cube structures which provide a solid foundation for effective and flexible analysis of the multidimensional ASRS text database. The text cube structure is constructed based on the TF/IDF (i.e., vector space) model while the topic cube is based on a probabilistic topic model. Techniques have also been developed for mining repetitive gapped subsequences [12], multi-concept document classification [48][49], and weakly supervised cause analysis [1]. The work in [30] has been further extended in [13] where the authors have proposed a keyword search technique. Given a keyword query, the algorithm ranks the aggregations of reports, instead of individual reports. For example, given a query "forced landing" an analyst may be interested in finding the external conditions (e.g. weather) that causes this kind of query and also find other anomalies that might co-occur with this one. This kind of analysis can be supported through keyword search, providing an analyst a ranked list of such aggregations for efficient browsing of relevant reports. In order to enrich the semantic information in a multidimensional text database for anomaly detection and causal analysis, Persing and Ng have developed new techniques for text mining and causal analysis from ASRS reports using semi-supervised learning [40] and subspace clustering [3].

Some work has also been done on categorizing ASRS reports into anomalous categories. It poses some specific challenges such as high and sparse dimensionality as well as multiple labels per document. Oza et al. [39] presents an algorithm called Mariana which learns a one-vs-all SVM classifier per anomaly category on the bag-of-words matrix. This provides good accuracy on most of the ASRS anomaly categories.

Topic detection from ASRS datasets have also received some recent attention. Shan et al. have developed the Discriminant Latent Dirichlet Allocation (DLDA) model [42], which is a supervised version of LDA. It incorporates label information into the generative model using logistic regression. Compared to Mariana, it not only has a better accuracy, but it also provides the topics along with the classification.

Gaussian Process Topic Models (GPTMs) by Agovic and Banerjee [2] is a novel family of topic models which define a Gaussian Process Mapping from the document space into the topic space. The advantage of GPTMs is that it can incorporate semi-supervised information in terms of a Kernel over the documents. It also captures correlations among topics, which leads to a more accurate topic model compared to LDA. Experiments on ASRS dataset show better topic detection compared to LDA. The experiments also illustrate that the topic space can be manipulated by changing the Kernel over documents.

3.3 Understanding categories

3.3.1 Recovering categories

In our first experiment, we sought to understand if the sparse learning methods could perform well in a blind test. The category data did not contain category *names*, only referring to them with letter capitals. We sought to understand what these categories were about. To this end, we have solved one LASSO problem for each category, corresponding to classifying that category against all the others. As shown in Table 1, we did recover a very accurate and differentiated image of the categories. For example, the categories M, T, U correspond to the ASRS categories *Weather/Turbulence*, *Smoke/Fire/Fumes/Odor,* and *Illness*. These categories names are part of the ASRS Events Categories as defined in `http://asrs.arc.nasa.gov/docs/dbol/ASRS_Database_Fields.pdf`. This blind test indicates that the method reveals the correct underlying categories using the words in the corpus alone.

The analysis reveals that there is a singular category, labelled B. This category makes up about 50% of the total number of reports. Its LASSO images points to two terms, which happen to be two categories, A (mechanical issues) and N (airspace issues). The other terms in the list are common to either A or N. The analysis points to the fact that category is a "catch-all" one, and that many reports in it could be re-classified as A or N.

Category	term 1	term 2	term 3	term 4	term 5	term 6	term 7
A (1441)	MEL	install	maintain	mechanic	defer	logbook	part
B (12876)	CATA	CATN	airspace	install	MEL	AN	
C (393)	abort	reject	ATO	takeoff	advance	TOW	pilot
D (428)	grass	CATJ	brake	mud	veer	damage	touchdown
E (3062)	runway	taxi	taxiway	hold	tower	CATR	ground control
F (6065)	CATH	clearance	cross	hold	feet	runway	taxiway
G (1684)	altitude	descend	feet	CATF	flightlevel	autopilot	cross
H (2213)	turn	head	course	CATF	radial	direct	airway
I (405)	knotindicator	speed	knot	slow	airspeed	overspeed	speedlimit
J (1107)	CATO	CATD	wind	brake	encounter	touchdown	pitch
K (353)	terrain	GPWS	GP	MD	glideslope	lowaltitude	approach
L (3357)	traffic	TACAS	RA	AN	climb	turn	separate
M (2162)	weather	turbulent	cloud	thunderstorm	ice	encounter	wind
N (1261)	airspace	TFR	area	adiz	classb	classdairspace	contact
O (325)	CATJ	glideslope	approach	high	goaraound	fast	stabilize
P (935)	goaround	around	execute	final	approach	tower	miss
Q (394)	gearup	land	towerfrequency	tower	contacttower	gear	GWS
R (1139)	struck	damage	bird	wingtip	truck	vehicle	CATE
S (6767)	maintain	engine	emergency	CATA	MEL	gear	install
T (647)	smoke	smell	odor	fire	fume	flame	evacuate
U (304)	doctor	paramedic	nurse	ME	breath	medic	physician
V (574)	police	passenger	behave	drink	alcohol	seat	firstclass

Table 1: LASSO images of the categories: each list of terms correspond to the most predictive list of features in the classification of one category against all the others. The numbers in parentheses denote the number of reports in each category. The meaning of abbreviations is listed in Table 2.

Meaning	Abbreviation	Meaning	Abbreviation
aborted take-off	ATO	minimumdescent	MD
aircraftnumber	AN	minimumequipmentlist	MEL
airtrafficcontrol	ATC	noticestoairspace	NTA
gearwarningsystem	GWS	resolutionadvisory	RA
groundproximity	GP	trafficalertandcollisionavoidancesystem	TACAS
groundproximitywarningsystem	GPWS	takeoffclear	TOC
groundproximitywarningsystemterrain	GPWS-T	takeoffwarning	TOW
knotsindicatedairspeed	KIAS	temporaryflightrestriction	TFR
medicalemergency	ME		

Table 2: Some abbreviations used in the ASRS data.

3.3.2 Sparse PCA for understanding categories

In this section, we plot the data set on a pair of axes that contain a lot of the variance, at the same time maintaining some level of interpretability to each of the four directions. Here the purpose is simply to perform an exploratory data analysis step, and evaluate if the results are consistent with domain knowledge. Our choice for setting the number of (sparse) principal components to two is not related to the data set itself. Rather, our choice simply allows us to plot the data on a two-dimensional figure, each component leading to one positive or negative direction.

We have proceeded with this analysis on the category data set. To this end we have applied a sparse PCA algorithm (power iteration with hard thresholding) to the category data matrix M (with each column an ASRS report), and obtained Fig. 1. We have not thresholded the direction q, only the direction p, which is the vector along which we project the points, so that it has at most 10 positive and 10 negative components. Hence, on our plot the underlying space is that corresponding to vector p. The sparse PCA plot shows that

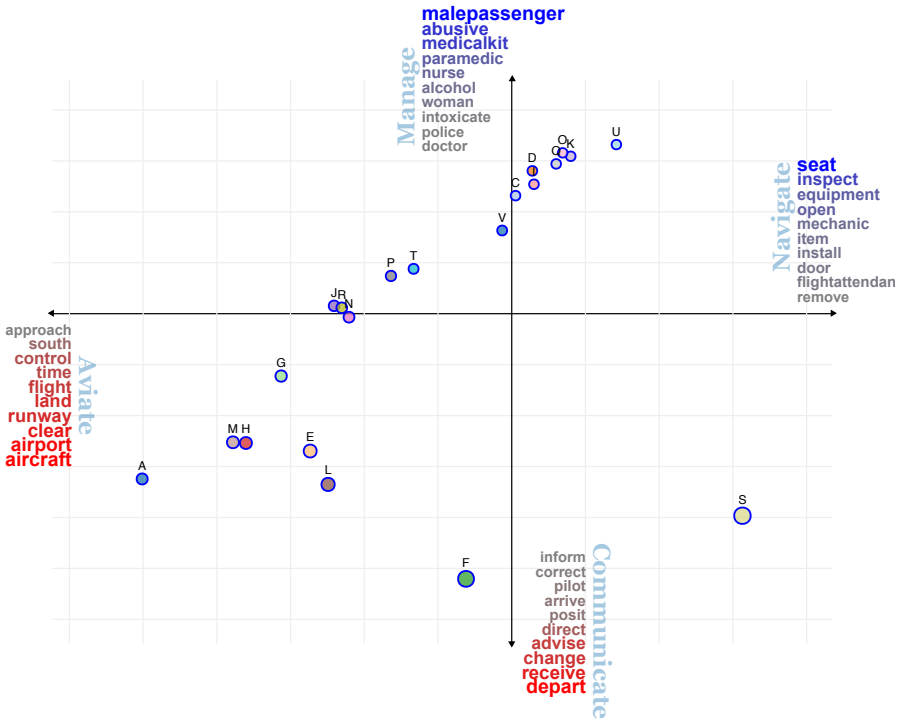

Figure 1: A sparse PCA plot of the category ASRS data. Here, each data point is a category, with size of the circles consistent with the number of reports in each category. We have focussed the axes and visually removed category B which appears to be a catch-all category. Each direction of the axes is associated with only a few terms, allowing an easy understanding of what each means. Each direction matches with one of the missions assigned to pilots in FAA documents (in light blue).

the data involves four different themes, each corresponding to the positive and negative directions of the first two sparse principal components.

Without any supervision, the sparse PCA algorithm found themes that are consistent with the four missions of pilots, as is widely cited in aviation documents [27]: *Aviate*, *Navigate*, *Communicate*, and *Manage Systems*. These four actions form the basis of flight training for pilots in priority order. The first and foremost activity for a pilot is to aviate, *i.e.*, ensure that the airplane stays aloft and in control. The second priority is to ensure that the airplane is moving in the desired direction with appropriate speed, altitude, and heading. The third priority is to communicate with other members of the flight crew and air traffic

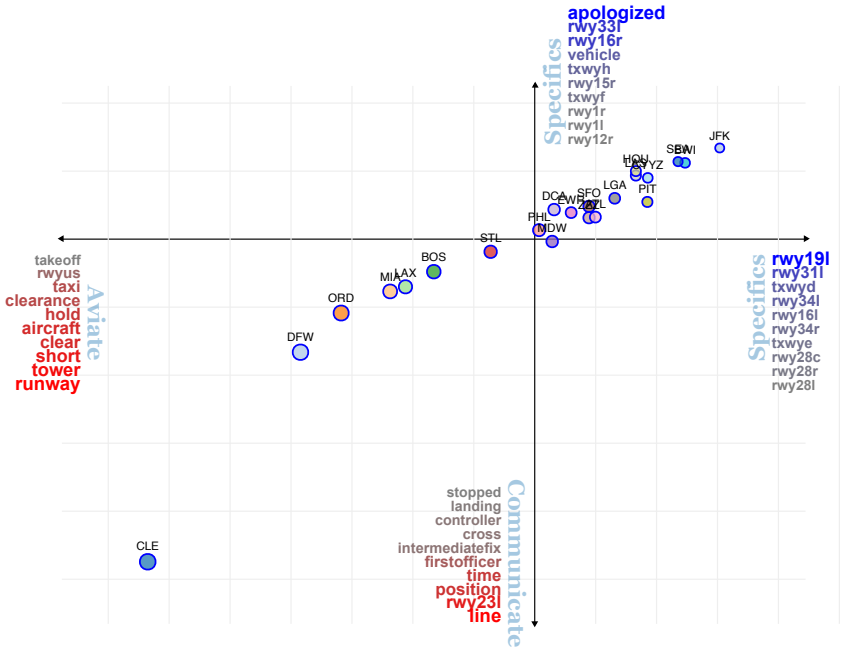

Figure 2: A sparse PCA plot of the runway ASRS data. Here, each data point is an airport, with size of the circles consistent with the number of reports for each airport.

control as appropriate. The final priority is to manage the systems (and humans involved) on the airplane to ensure safe flight. These high-level tasks are critical for pilots to follow because of their direct connection with overall flight safety.

The sparse algorithm discovers these four high-level tasks as the key factors in the category data set. The words associated with each direction in Fig. 1 (for example, "seat". "inspect", etc, along the East direction) were automatically assigned by the algorithm. On the plot, we manually assigned a higher-level label (such as "Navigate") to the list of words associated with each direction. As claimed, the list of words are very consistent with the high-level labels.

We validated our discovery by applying the Latent Dirichlet Allocation algorithm to the ASRS data and set the desired number of topics equal to 4. Because there is currently no method to discover the 'correct' number of topics, we use this high-level task breakdown as for an estimate of the number of topics described in the documents. While the results did not reveal the same words as sparse PCA, it revealed a similar task breakdown structure. More detailed results involving LDA are described in section **??**.

In a second illustration we have analyzed the runway data set described in Appendix A. Fig 2 shows that two directions remain associated with the themes found in the category data set, namely "aviate" (negative horizontal direction) and "communicate". The airports near those directions, in the bottom left quadrant of the plot (CLE, DFW, ORD, LAX, MIA, BOS) are high-traffic ones with relatively bigger number of reports, as is indicated by the size of the circles. This is to be expected from airports where large amounts of communication is necessary (due to high traffic volume and complicated layouts). Another cluster (on the NE quadrant) corresponds to the two remaining directions, which we labelled "specifics" as they related to specific runways and taxiways in airports. This other cluster of airports seem to be affected by issues related to specific runway configuration that are local to each airport.

In a second plot (Fig. 3) we redid the analysis after removal of all the features related to runways and taxiways, in order to discover what is "beyond" runway and taxiway issues. We recover the four themes of *Aviate*, *Navigate*, *Communicate* and *Manage*. As before, high-traffic airports remain affected mostly by

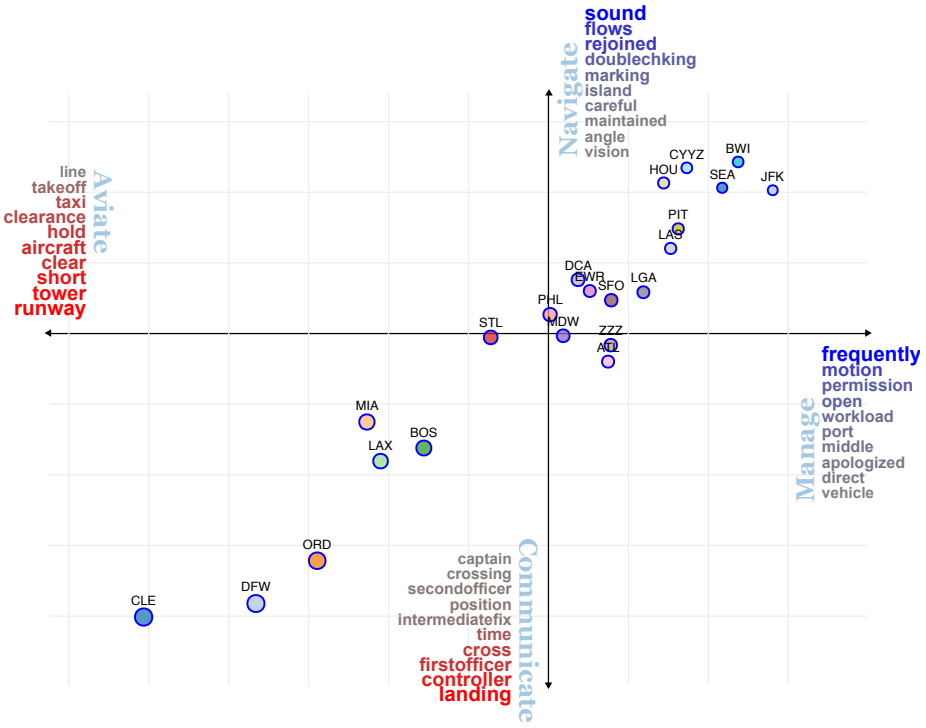

Figure 3: A sparse PCA plot of the runway ASRS data, with runway features removed.

aviate and communicate issues. Note that the disappearance of passenger-related issues within the *Manage* theme, which was defining the positive-vertical direction in Fig 1. This is to be expected, since the data is now restricted to runway issues: what involved passenger issues in the category data set, now becomes mainly related to the other humans in the loop, pilots ("permission"), drivers ("vehicle") and other actors, and their actions or challenges ("workload, open, apologized").

A look at the sparse PCA plots (Figs. 3 and 1) reveals a commonality: the themes of *Aviate* and *Communicate* seem to go together in the data, and are opposed to the other sub-group of *Navigate* and *Manage Systems*.

How about thresholded PCA? Fig. 4 shows the total explained variance by the two methods (sparse and thresholded PCA) as a function of the number of words allowed for the axes, for the category data set. We observe that thresholded PCA does not explain as much variance (in fact, only half as much) as sparse PCA, with the same budget of words allowed for each axis. This ranking is reversed only after 80 words are allowed in the budget. The two methods do reach the maximal variance explained by PCA as we relax our word-budget constraint. Similar observations can be made for the runway data set.

3.3.3 Thresholded LDA

For the sake of comparison, we have also applied the Latent Dirichlet Allocation (LDA) algorithm to the ASRS data. LDA is an unsupervised technique for topic modeling and as such it requires the number of topics to extract from the data. For our ASRS data, we have generated 4, 6, and 10 topics. We have used the code in [45], with default parameter values, as detailed in Table 3.

In a first attempt, we have not removed any stop words and found the corresponding lists to be quite uninformative, as stop words did show up. We have then removed the stop words using a standard list of

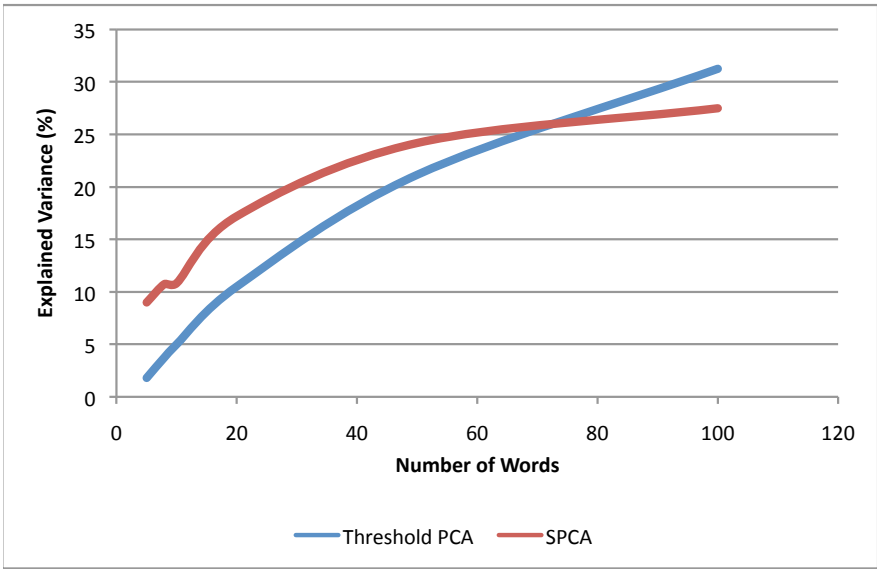

Figure 4: Explained variance.

```
NUMTOPICS=10
BETA=0.01;
ALPHA=50/NUMTOPICS;
ITERATIONS = 500; (LDA iterations)
WORDSPERTOPIC = 10;
SEED = 1000; (used for Gibbs sampler initialization)
```

Table 3: Table of default parameters used in the LDA code of [45].

stop words from the English dictionary[2].

Table 4 shows the four topics with the top 10 words (according to posterior distribution) thresholded from the entire distribution of words. Unlike the Spase PCA method (Fig. 1), the 4 topics of LDA model do not correspond to the four missions of the pilot: *Aviate*, *Navigate*, *Communicate*, and *Manage Systems*. In fact, there are certain words such as 'aircraft', 'runway' etc. which seem to occur in most of the topics and are therefore not very informative for discrimination purposes. From a high level, the topics roughly seem to correspond to the following: (1) Topic 1 – gate events or ground events, (2) Topic 2 – ATC communication or clearance related, (3) Topic 3 – not clear , and (4) Topic 4 – approach/landing.

Tables 5 and 6 depicts 6 and 10 topics extracted from the ASRS data. Both of these tables show that there the topics are not very unique since the words in the topics appear to be substantially overlapping and therefore, (1) there is not much discriminative power of the components (words) and, (2) the topics do not discover unique structures in the data. Finally, we report the running time of LDA algorithm for these three

[2]The list can be consulted at `http://www.eecs.berkeley.edu/~gawalt/MIR2010/NYTWStops.txt`.

Topic	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8	term 9	term 10
1	runway	taxi	ground	taxiway	turn	control	captain	airport	gate	txwyno
2	tower	clear	takeoff	aircraft	rwyus	clearance	position	firstofficer	captain	flight
3	runway	hold	short	line	cross	told	rwyus	nar	aircraft	taxi
4	runway	aircraft	landing	ctlapproachcontrol	feet	report	lights	pilot	due	crew

Table 4: 4 topics extracted from ASRS dataset.

experiments: 12 secs for 4 topics, 16 secs for 6 topics and 25 secs for 10 topics. For all these experiments, we have run the gibbs sampler for 500 iterations.

Topic	1	2	3	4	5	6
term 1	captain	runway	runway	tower	runway	runway
term 2	firstofficer	hold	taxi	clear	aircraft	taxiway
term 3	time	short	ground	takeoff	landing	taxi
term 4	flight	line	control	clearance	ctlapproachcontrol	turn
term 5	departure	aircraft	cross	position	feet	airport
term 6	secondofficer	stopped	rwyus	rwyus	traffic	report
term 7	airtrafficcontrol	taxi	instructions	aircraft	tower	txwyno
term 8	chklist	stop	crossing	aircarrier	clear	lights
term 9	txwyme	nar	told	controller	landed	end
term 10	crew	clear	gate	call	approximately	area

Table 5: 6 topics extracted from ASRS dataset.

3.4 Analysis of runway incursion incidents

In this section, our objective is to understand specific runway-related issues affecting each airport using the runway ASRS data.

We will use three different methods to obtain the image (as given by a short list of terms) for each airport. A first approach is basic and relies on co-occurrence between the airport's name and the other terms appearing in documents mentioning that name. The two other approaches, thresholded naïve Bayes and LASSO, rely on classification. For this, we separate the data into two sets: one set corresponds to the ASRS reports that contain the name of the airport under analysis; the other contains all the remaining ASRS documents in our corpus. We have selected for illustration purposes the top twenty airports, as ordered by the number of reports that mention their name.

3.4.1 Co-occurrence analysis

With no stop words removed or word-stemming, the simplest method is the co-occurence on term frequency, which expectedly gives commonly-used words with little meaning as term association for the airports. Results are shown in Table 7. Among these top words across the airports are simply "the", "runway", "and".

We also experiment with the TF-IDF scores for the co-occurence method, which adds a weight of inverse document frequency to each term. When considering an airport, TF-IDF generally favors terms that occur more exclusively in documents containing the name of that airport. Results are shown in Table 8. Among the top 8 terms chosen for each airport in the experimentation are: the airport name (ATL, LGA, LAS, BWI, JFK) and specific runways with taxiways that have reported aviation issues. Some focus on actions are shown in a few airports: MIA (takeoff), PHL (cross), DCA and BWI (turn).

3.4.2 Naïve Bayes classification

To emphasize the differences between two sets of documents, one method is to make use of the Naïve Bayes classifier on the binary term-appearance matrix. This method relies on a strong assumption of term's independence across the whole corpus. To obtain the term association for each airport, we compute the estimated log-odds ratio of term appearance in "positive" documents to that in "negative" ones, normalized by the variance of this estimation, in order to cope with noise in the data. Hard thresholding these log-odds ratios allows to retain a fixed number of terms associated to each airport. Results from the Naïve Bayes classification are shown in Table 9. It seems that the method, applied to the runway ASRS dataset, is effective in pointing out generic actions relevant to the aviation system. Term associations mostly reveal "cross", "landed", "tower" as strong discriminating features. Nevertheless, this generic result provides little help in understanding specific runway-related issues that affect each airport.

Topic	1	2	3	4	5
term 1	runway	runway	aircraft	taxi	runway
term 2	clearance	taxiway	runway	time	ground
term 3	controller	taxi	tower	airport	taxi
term 4	cross	end	clear	ramp	rwyus
term 5	clear	txwyno	takeoff	crew	control
term 6	crossing	lights	landing	flight	told
term 7	back	airport	rwyus	departure	instructions
term 8	instructions	turn	nar	problem	gate
term 9	airtrafficcontrol	turned	stop	due	instructed
term 10	aircraft	side	speed	factors	crossed

Topic	6	7	8	9	10
term 1	tower	hold	runway	ctlapproachcontrol	captain
term 2	takeoff	short	intermediatefix	feet	taxi
term 3	position	line	txwyme	landing	firstofficer
term 4	clear	runway	txwyno	aircraft	runway
term 5	rwyus	report	secondofficer	traffic	turn
term 6	aircarrier	stopped	intersection	final	chklist
term 7	call	past	asked	approximately	time
term 8	frequency	nar	txwydo	land	looked
term 9	heard	taxi	txwygo	report	rwy4l
term 10	clearance	holding	approach	pilot	txwyb

Table 6: 10 topics extracted from ASRS dataset.

airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	the	runway	and	i	was	hold	short	a
DFW	the	runway	and	i	was	tower	a	aircraft
ORD	the	runway	and	i	was	a	that	were
MIA	the	runway	and	was	i	a	hold	taxi
BOS	the	runway	and	i	was	a	hold	were
LAX	the	runway	and	i	was	a	hold	short
STL	the	runway	and	i	was	short	a	that
PHL	the	runway	and	was	i	aircraft	taxi	a
MDW	the	runway	and	i	was	a	taxi	hold
DCA	the	runway	and	i	was	a	were	that
SFO	the	runway	and	i	was	a	that	aircraft
ZZZ	the	and	runway	i	was	a	aircraft	were
EWR	the	runway	and	i	was	a	tower	that
ATL	the	runway	and	was	i	a	aircraft	tower
LGA	the	runway	and	was	i	aircraft	hold	a
LAS	the	runway	and	i	was	a	for	were
PIT	the	runway	and	was	i	a	taxi	that
HOU	the	runway	and	i	was	for	a	rwy12r
BWI	the	runway	and	was	i	taxi	a	that
CYYZ	the	runway	and	hold	short	was	i	line
SEA	the	runway	and	i	was	hold	tower	a
JFK	the	runway	and	was	i	a	that	clear

Table 7: Images of airports via the co-occurrence method on the binary term by document matrix, without stop word removal.

airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	rwy23l	rwy24l	rwy24c	cle	rwy23r	rwy5r	rwy6r	rwy5l
DFW	rwy18l	dfw	rwy17r	rwy35l	rwy35c	rwy17c	rwy18r	rwy36r
ORD	ord	rwy22r	rwy27r	rwy32r	rwy27l	rwy9l	rwy4l	rwy22l
MIA	rwy9l	mia	txwyq	rwy9r	line	rwy8r	txwym	takeoff
BOS	rwy4l	bos	rwy33l	rwy22r	rwy22l	rwy4r	captain	frequency
LAX	rwy25r	lax	rwy25l	rwy24l	rwy24r	i	captain	firstofficer
STL	rwy30l	rwy12l	rwy12r	stl	rwy30r	cross	aircarrier	short
PHL	rwy9l	rwy27r	phl	rwy27l	txwyk	x	e	cross
MDW	rwy31c	rwy31r	mdw	rwy22l	rwy4r	txwyp	midway	rwy13c
DCA	dca	txwyj	airplane	turn	ground	traffic	i	pad
SFO	rwy28l	rwy28r	sfo	rwy1l	rwy1r	rwy10r	rwy10l	captain
ZZZ	xxr	zzz	radio	hangar	tow	i	speed	rwyxa
EWR	rwy4l	rwy22r	ewr	rwy22l	txwyp	txwyz	rwy4r	txwypb
ATL	atl	rwy26l	rwy8r	rwy9l	rwy27r	rwy26r	dixie	atlanta
LGA	lga	txwyb	instrumentlandingsystem	txwyb4	vehicle	line	lights	txwyp
LAS	las	rwy25r	rwy19l	rwy7l	rwy1r	rwy19r	rwy25l	rwy1l
PIT	rwy28c	rwy10c	pit	rwy28l	txwye	txwyw	txwyv	txwyn1
HOU	rwy12r	hou	rwy12l	heading	takeoff	i	rwy30r	txwyme
BWI	bwi	rwy15r	txwyp	rwy33l	turn	intersection	txwyp1	taxiway
CYYZ	txwyq	txwyh	yyz	line	rwy6l	rwy33r	short	length
SEA	rwy34r	rwy16l	rwy34l	sea	rwy16r	position	firstofficer	y
JFK	jfk	rwy31l	vehicle	rwy13r	rwy4l	rwy22r	rwy13l	rwy31r

Table 8: Images of airports via the co-occurrence method, using TF-IDF scores.

airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	line	short	this	are	for	following	first	didn
DFW	cross	crossing	tower	landed	aircraft	across	short	holding
ORD	turn	but	when	l	speed	get	than	txwyb
MIA	rwy9l	txwyp	taxiway	txwym	signage	chart	line	via
BOS	frequency	told	s	contact	controller	txwyk	rwy4l	rwy22r
LAX	tower	cross	short	call	high	landing	speed	firstofficer
STL	cross	line	short	call	hold	aircraft	trying	supervisor
PHL	cross	e	rwy9l	crossed	x	txwye	spot	txwyk
MDW	clearance	i	taxi	hold	gate	captain	crossed	short
DCA	his	turn	just	captain	but	airplane	txwyj	through
SFO	control	crossing	short	crossed	some	txwyb	landing	cross
ZZZ	radio	time	proceeded	while	way	any	i	approximately
EWR	tower	landing	aircraft	txwyp	rwy22r	between	high	rwy4l
ATL	cross	crossing	roll	speed	high	hold	txwyd	knot
LGA	txwyb	aircarrier	instrumentlandingsystem	off	lights	txwyp	behind	error
LAS	after	saw	rwy25r	procedure	lights	approximately	never	signs
PIT	via	txwye	intersection	firstofficer	conversation	night	looking	down
HOU	txwyme	hold	rwy12r	takeoff	via	around	trying	little
BWI	turn	intersection	taxi	mistake	ground	gate	made	crossed
CYYZ	line	short	stopped	hold	past	taxi	full	end
SEA	feet	firstofficer	cross	tower	read	after	called	back
JFK	prior	instructed	departure	report	his	being	out	txwya

Table 9: Images of airports via Naïve Bayes classification, using the binary term by document data.

airport	term 1	term 2	term 3	term 4	term 5	term 6	term 7	term 8
CLE	Rwy23L	Rwy24L	Rwy24C	Rwy23R	Rwy5R	Line	Rwy6R	Rwy5L
DFW	Rwy35C	Rwy35L	Rwy18L	Rwy17R	Rwy18R	Rwy17C	cross	Tower
ORD	Rwy22R	Rwy27R	Rwy32R	Rwy27L	Rwy32L	Rwy22L	Rwy9L	Rwy4L
MIA	Rwy9L	TxwyQ	Rwy8R	Line	Rwy9R	PilotInCommand	TxwyM	Takeoff
BOS	Rwy4L	Rwy33L	Rwy22R	Rwy4R	Rwy22L	TxwyK	Frequency	Captain
LAX	Rwy25R	Rwy25L	Rwy24L	Rwy24R	Speed	cross	Line	Tower
STL	Rwy12L	Rwy12R	Rwy30L	Rwy30R	Line	cross	short	TxwyP
PHL	Rwy27R	Rwy9L	Rwy27L	TxwyE	amass	TxwyK	AirCarrier	TxwyY
MDW	Rwy31C	Rwy31R	Rwy22L	TxwyP	Rwy4R	midway	Rwy22R	TxwyY
DCA	TxwyJ	Airplane	turn	Captain	Line	Traffic	Landing	short
SFO	Rwy28L	Rwy28R	Rwy1L	Rwy1R	Rwy10R	Rwy10L	b747	Captain
ZZZ	hangar	radio	Rwy36R	gate	Aircraft	Line	Ground	Tower
ERW	Rwy22R	Rwy4L	Rwy22L	TxwyP	TxwyZ	Rwy4R	papa	TxwyPB
ATL	Rwy26L	Rwy26R	Rwy27R	Rwy9L	Rwy8R	atlanta	dixie	cross
LGA	TxwyB4	ILS	Line	notes	TxwyP	hold	vehicle	Taxiway
LAS	Rwy25R	Rwy7L	Rwy19L	Rwy1R	Rwy1L	Rwy25L	TxwyA7	Rwy19R
PIT	Rwy28C	Rwy10C	Rwy28L	TxwyN1	TxwyE	TxwyW	Rwy28R	TxwyV
HOU	Rwy12R	Rwy12L	citation	Takeoff	Heading	Rwy30L	Line	Tower
BWI	TxwyP	Rwy15R	Rwy33L	turn	TxwyP1	Intersection	TxwyE	Taxiway
CYYZ	TxwyQ	TxwyH	Rwy33R	Line	YYZ	Rwy24R	short	toronto
SEA	Rwy34R	Rwy16L	Rwy34L	Rwy16R	AirCarrier	FirstOfficer	TxwyJ	SMA
JFK	Rwy31L	Rwy13R	Rwy22R	Rwy13L	vehicle	Rwy4L	amass	Rwy31R

Table 10: Images of airports via LASSO regression, using TF-IDF data.

3.4.3 LASSO

We turn to a LASSO regression to analyze the image of each airport. Our results, shown in Table 10, are based on the TF-IDF representation of the text data. They indicate that the LASSO images for each airport reveal runways that are specific to that airport, as well as some specific taxiways. We elaborate on this next.

3.4.4 Tree images via two-stage LASSO

To further illustrate the LASSO-based approach, we focus on a single airport (say DFW). We propose a two-stage LASSO analysis allowing to discover a tree structure of terms. We first run a LASSO algorithm to discover a short list of terms that correspond to the image of the term "DFW" in the data set. For each term in that image, we re-run a LASSO analysis, comparing all the documents in the DFW-related corpus containing the term, against all the other documents in the DFW-related corpus. Hence the second step in this analysis only involves the ASRS reports that contain the term "DFW". The approach produces a tree-like structure that can be visualized as two concentric circles of terms, as in Figs. 5 and 6.

The tree analysis, which is visualized in Figs. 5 and 6, highlights which issues are pertaining to specific runways, and where *attention* could be focussed. In the airport diagram in Figure 7, we have highlighted some locations discussed next.

As highlighted in red in the airport diagram 7, the major runway 35L crosses the taxiway EL; likewise for runway 36R and its siblings taxiway WL and F. Runway/taxiway intersections are generally known to contain a risk of collision. At those particular intersections, the issues seem to be about obtaining "clearance" to "turn" from the tower, which might be due to the absence of line of sight from the tower (here we are guessing that the presence of the west cargo area could be a line-of-sight hindrance). The corresponding tree image in Fig. 5 is consistent with the location of DFW in the sparse PCA plot (Fig. 3), close to the themes of *Aviate* and *Communicate*.

4 Sparse PCA and LDA: Comparative Study

In this section, we perform a comparative study of the sparse PCA and LDA approaches, using databases that are commonly used in the text processing community. This will help further illustrate the respective merits of the two methods for data sets other than the previously used ASRS data. We use three data sets, of increasing size: the Amazon data set, which contains consumer reviews for a variety of products;

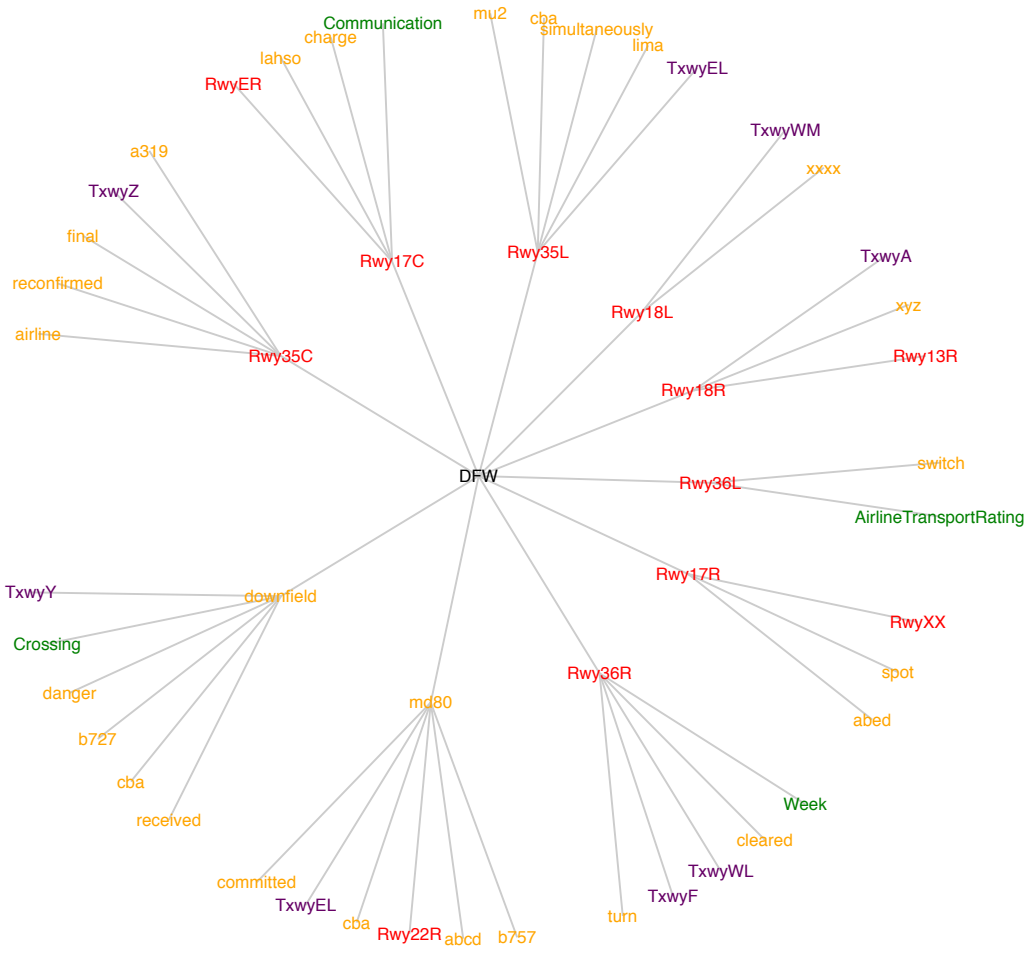

Figure 5: A tree LASSO analysis of the DFW airport, showing the LASSO image (inner circle) and for each term in that image, a further image.

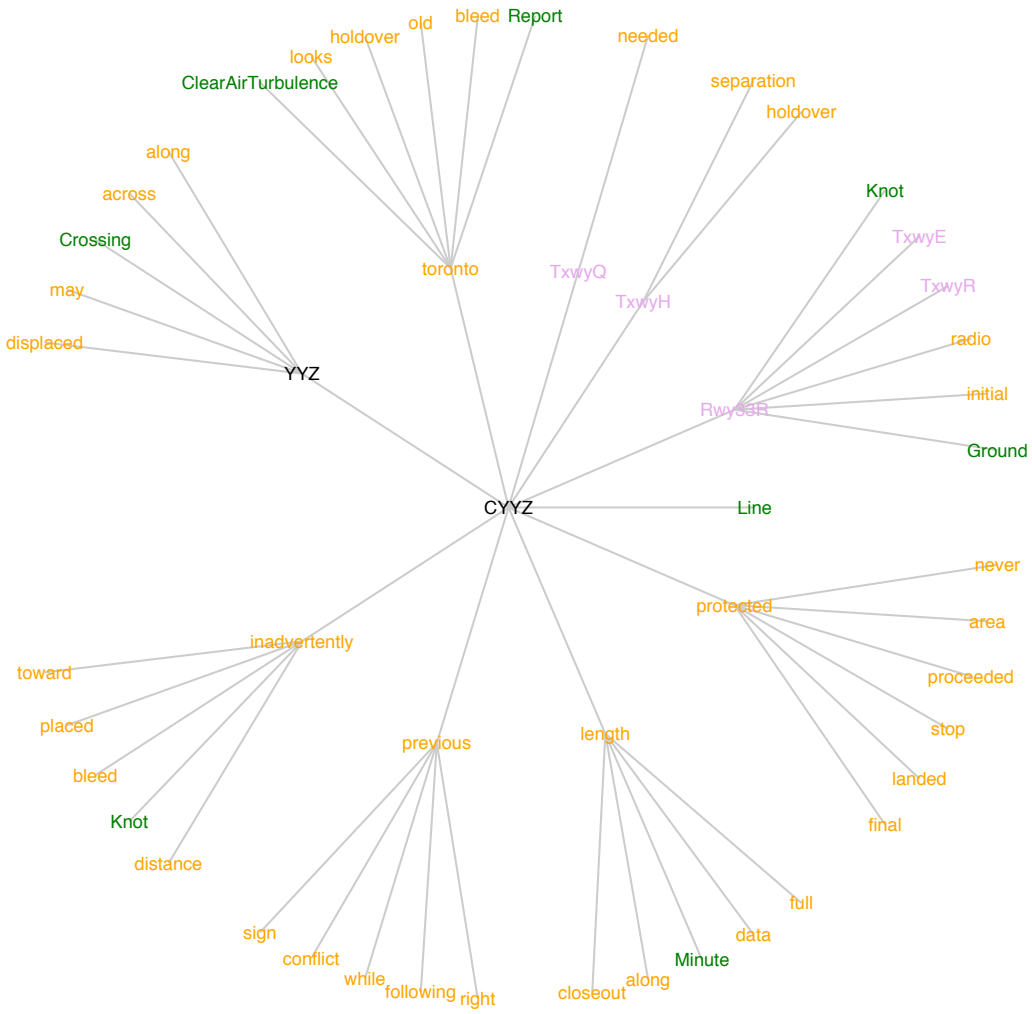

Figure 6: A tree LASSO analysis of the CYYZ airport, showing the LASSO image (inner circle) and for each term in that image, a further image.

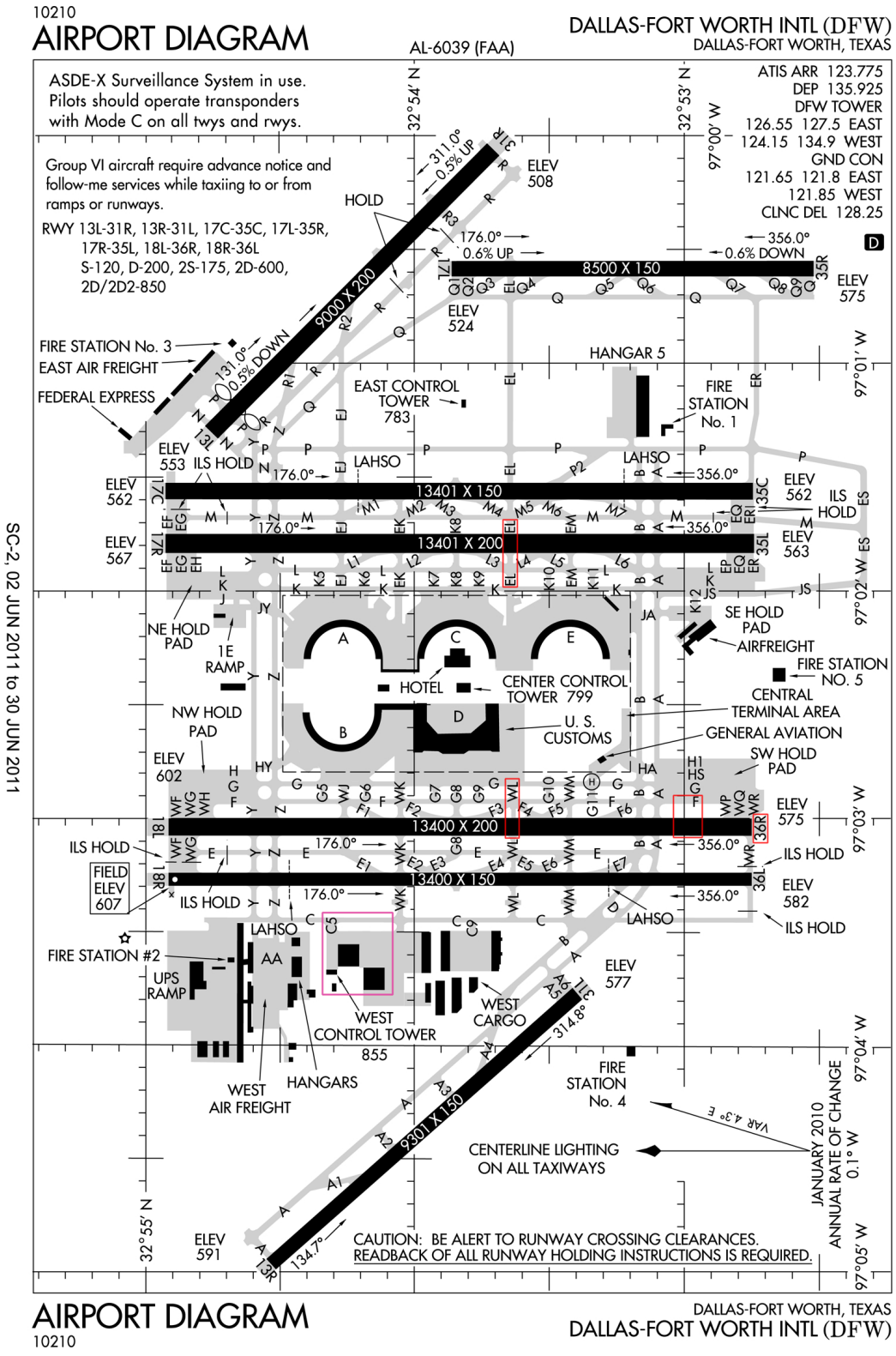

Figure 7: Diagram of DFW.

the Reuters news text categorization collection, which involves news articles; and the NSF data set, which contains abstracts from scientific articles dated 1999 through 2003. The three data sets can be obtained from the UCI archive [20]. These data sets range widely in size, from one to a thousand hundred documents. For each data set, we apply a very basic stop-word removal from a list of approximately 550 stop words[3].

With the ASRS data sets one of our goals in using sparse PCA was to plot the documents in two dimensions, hence we have selected two principal components. In this present study, we increase that number to 10 principal components and rigorously compare with the 10 topics revealed by LDA. The LDA code we have used has been developed by Steyvers and Griffiths [45]; throughout, we have used the default values for various parameters, as detailed in Table 3.

4.1 Amazon data set

The Amazon data set is the smallest of the three data sets examined in this section, with 1500 documents, and 2960 single words. It consists of user reviews, mainly on consumer products. The reviews originate from 50 of the most active users, each of whom has 30 reviews collected. The original data contains bigrams and trigrams, and also includes authors' usage of digits and punctuation. We have removed all of these, and retained only unigrams after stop-word removal to run the LDA and the SPCA.

The results are shown in Table 11. For both methods, the topics show very clear word associations. In these topics, when we rank words in non-increasing order of their weights, the topic words show up as top words. For SPCA, almost all topics are very easy to interpret: topic 1 corresponds to books, topic 2 to movies, topic 4 to games, topic 6 to cells and batteries, topic 7 to hair products, topic 8 to music. Topic 9 is less clear, but likely to be about electronic reading devices. For LDA, we see that topic 10 corresponds to books, topic 9 to stories, topic 3 to movie and topic 1 to music. LDA shows a similar good performance, although we see some non-informative words such as "good" appear in the lists.

4.2 Reuters data set

The Reuters 21578 data set contains 19043 documents and 38361 unique words. It is one of the most frequently used for text processing on news since 1999. The dataset is divided into several categories. However for the purpose of this study we have discarded the labels and any categorical information, treating all the documents on equal basis. Our goal here is to ascertain if sparse learning methods can handle data that is complex by the variety of topics, as well as the presence of acronyms and abbreviations. The results of LDA and SPCA are shown in Table 12.

For both methods, topics in the Reuters dataset are sometimes difficult to recognize, which is perhaps due to the complexity of this data set. There are topics that both the LDA and the SPCA agree upon. For example, LDA's topic 2 and SPCAs topic 1 have similar words: "mln" (million), "dlrs" (dollars), "net", "loss", "profit", "year" and "sales". LDA's topic 9 and SPCA's topic 5 are both on agriculture exports and oil/gas prices (with terms such as "wheat", "export", "tonnes", "price"). LDA's topic 7 and SPCA's topic 9 both discuss US government issues, with terms such as "President Reagan" and "John Roberts", respectively. Of the remaining topics, the two methods either share some commonalities (for example the LDA and SPCA topic 8 can both be guessed to be related to the European zone) or involve different topics (for example LDA's topic 1 is on economic issues, while SPCA topic 3 is on market exchange).

4.3 NSF data set

The NSF datasets contain a collection of abstracts of scientific papers, written between 1990 to 2003. This is the largest data set in our study, with over $120,000$ documents and over $30,000$ words. The results of LDA and SPCA are shown in Table 13.

In Table 14 we have summarized our interpretation of each topic, mostly based on the first (most heavily weighted) term for each method. (We deviated from the rule when the other terms were consistently pointing to a more specific topic, such as topic 8 for LDA or 5 for sparse PCA.)

[3]Available at `http://atticus.berkeley.edu/guanchengli/stopword.txt`.

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
love	0.02461	case	0.01669	film	0.03878	time	0.05518	product	0.02773
music	0.02357	light	0.01298	movie	0.02676	long	0.02607	easy	0.021
year	0.01749	included	0.01247	man	0.01594	recommend	0.02211	quality	0.0209
sound	0.01655	problem	0.01226	stars	0.01273	day	0.02171	make	0.01692
beautiful	0.01383	works	0.01216	american	0.01223	makes	0.0209	bit	0.01641
cd	0.01278	video	0.01082	bad	0.01042	fun	0.01765	hand	0.01539
great	0.01267	cells	0.01061	wife	0.01032	feel	0.01664	top	0.01346
fine	0.01267	system	0.01051	past	0.00982	good	0.01643	color	0.01305
art	0.01267	cable	0.01051	school	0.00972	game	0.01633	amazon	0.01295
christmas	0.01246	time	0.01041	films	0.00912	thing	0.01552	high	0.01193
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
good	0.026	good	0.02344	find	0.03938	story	0.04881	book	0.14368
dvd	0.01856	nice	0.01863	people	0.03684	stories	0.02108	read	0.03378
made	0.01834	set	0.01824	work	0.03633	life	0.01953	author	0.02054
series	0.01618	back	0.01755	found	0.02646	family	0.01797	books	0.01952
show	0.01586	small	0.01569	make	0.02595	young	0.01578	reading	0.01926
short	0.01554	great	0.01559	things	0.01618	children	0.01551	life	0.01765
version	0.01543	easily	0.01432	part	0.01598	years	0.01533	history	0.01426
style	0.01456	put	0.01402	thought	0.01343	characters	0.01478	written	0.01392
back	0.01359	buy	0.01324	information	0.0113	world	0.01277	reader	0.01256
set	0.01349	pretty	0.01285	making	0.01099	TRUE	0.01058	interesting	0.01188

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
book	0.9127	film	0.6622	nice	0.3493	game	0.7527	skin	0.5492
read	0.1527	movie	0.3617	side	0.3464	games	0.3034	children	0.3864
good	0.1437	product	0.2666	lot	0.3042	fun	0.294	young	0.3028
story	0.1331	set	0.2268	price	0.2896	play	0.2388	man	0.2378
time	0.1228	made	0.2051	light	0.287	family	0.1969	written	0.2334
life	0.1207	years	0.2014	day	0.275	world	0.1619	dry	0.2098
author	0.1058	makes	0.1889	place	0.2703	characters	0.1572	beautiful	0.208
find	0.1016	long	0.1633	series	0.2688	level	0.1477	case	0.2071
people	0.1008	dvd	0.1573	works	0.2354	character	0.1275	feel	0.2044
reading	0.0867	back	0.1566	small	0.2329	played	0.1062	times	0.1997
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
cells	0.7011	hair	0.8361	songs	0.4754	cover	0.8295	writing	0.5158
capacity	0.3149	recommended	0.2397	album	0.4717	wife	0.2398	products	0.4092
mah	0.2537	style	0.2042	christmas	0.3894	similar	0.1887	handle	0.3363
nimh	0.2503	brush	0.2002	cd	0.3492	told	0.1881	perfect	0.274
aa	0.2351	highly	0.18	voice	0.2577	purchased	0.187	material	0.2682
aaa	0.2276	plastic	0.1695	song	0.2347	avoid	0.162	desk	0.2139
charger	0.1924	expensive	0.149	track	0.2011	practical	0.162	short	0.2031
package	0.1769	put	0.1252	fan	0.147	paid	0.1532	color	0.2007
cell	0.1751	hold	0.1148	fine	0.1313	kindle	0.1431	lines	0.2005
rechargeable	0.1511	ingredients	0.1054	hear	0.1247	history	0.1061	review	0.1638

Table 11: Comparison between LDA (top) and Sparse PCA (bottom) on the Amazon data set.

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
project	0.01809	species	0.01644	research	0.09839	theory	0.02179	materials	0.01826
data	0.01516	study	0.01247	university	0.04056	problems	0.0205	high	0.01393
research	0.01218	important	0.00828	program	0.02237	methods	0.01438	properties	0.0127
information	0.00947	natural	0.00822	award	0.01927	study	0.01279	phase	0.01165
social	0.00909	provide	0.00742	chemistry	0.01804	work	0.01037	chemical	0.00837
study	0.00855	evolution	0.00721	support	0.01735	systems	0.01011	surface	0.00796
model	0.00832	understanding	0.00692	state	0.0148	problem	0.00912	energy	0.00787
models	0.00684	patterns	0.00679	dr	0.01355	mathematical	0.00892	optical	0.00747
economic	0.00597	environmental	0.00638	equipment	0.01101	models	0.00891	magnetic	0.00736
understanding	0.00573	studies	0.00604	project	0.01023	analysis	0.00799	electron	0.0068
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
research	0.04002	molecular	0.01428	data	0.01493	students	0.04399	system	0.01947
scientists	0.01163	cell	0.01152	water	0.01087	science	0.03092	design	0.01944
national	0.01114	protein	0.01101	processes	0.00908	project	0.02327	systems	0.01931
researchers	0.01018	specific	0.01058	study	0.00893	program	0.01754	control	0.01329
workshop	0.00976	function	0.00979	model	0.00772	engineering	0.01521	based	0.01257
scientific	0.00958	cells	0.00942	flow	0.00761	education	0.01385	performance	0.01067
field	0.00911	studies	0.00894	ocean	0.00748	undergraduate	0.0127	data	0.01043
support	0.009	proteins	0.0089	climate	0.0067	laboratory	0.01098	develop	0.00939
areas	0.00894	mechanisms	0.00883	ice	0.00629	faculty	0.01078	network	0.00839
international	0.00886	dna	0.00804	field	0.00622	learning	0.0107	software	0.00814

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
mln	0.5898	common	0.0623	government	0.1019	current	0.069	sources	0.0935
cts	0.5794	payable	0.0652	oil	0.1073	ended	0.0768	compared	0.0971
net	0.3028	bank	0.0685	agreement	0.1093	extraordinary	0.0812	price	0.1035
shr	0.2863	july	0.0695	president	0.1127	sale	0.0855	fell	0.1069
dlrs	0.1987	share	0.0716	due	0.1369	credit	0.095	production	0.1088
loss	0.1853	june	0.0839	trade	0.1415	discontinued	0.1079	prices	0.1097
revs	0.1738	corp	0.0992	debt	0.1558	operations	0.1395	exports	0.1142
profit	0.0922	shares	0.1055	today	0.1651	includes	0.1695	department	0.1187
year	0.0722	stock	0.115	york	0.191	excludes	0.2029	export	0.1214
sales	0.0719	company	0.1355	offering	0.2031	gain	0.2119	total	0.1395
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
years	0.0903	rates	0.076	eurobond	0.1496	john	0.0549	directors	0.0743
spokesman	0.1008	provided	0.0837	luxembourg	0.1549	robert	0.0596	paid	0.0871
air	0.1132	week	0.0843	listed	0.1586	subsidiary	0.0605	outstanding	0.0879
work	0.118	interest	0.0845	denominations	0.1735	american	0.0609	increase	0.0927
division	0.1195	central	0.0851	underwriting	0.188	elected	0.0623	offer	0.0952
awarded	0.1244	estimate	0.0951	selling	0.2023	director	0.0739	initial	0.0979
federal	0.1495	forecast	0.0969	issuing	0.2074	effective	0.0811	cash	0.1159
general	0.154	revised	0.1189	payment	0.214	resigned	0.0864	approved	0.1313
expected	0.1588	assistance	0.1208	date	0.2334	financial	0.1265	annual	0.144
international	0.2345	shortage	0.124	management	0.2385	operating	0.1584	meeting	0.1544

Table 12: Comparison between LDA (top) and Sparse PCA (bottom) on the Reuters data set.

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
project	0.01809	species	0.01644	research	0.09839	theory	0.02179	materials	0.01826
data	0.01516	study	0.01247	university	0.04056	problems	0.0205	high	0.01393
research	0.01218	important	0.00828	program	0.02237	methods	0.01438	properties	0.0127
information	0.00947	natural	0.00822	award	0.01927	study	0.01279	phase	0.01165
social	0.00909	provide	0.00742	chemistry	0.01804	work	0.01037	chemical	0.00837
study	0.00855	evolution	0.00721	support	0.01735	systems	0.01011	surface	0.00796
model	0.00832	understanding	0.00692	state	0.0148	problem	0.00912	energy	0.00787
models	0.00684	patterns	0.00679	dr	0.01355	mathematical	0.00892	optical	0.00747
economic	0.00597	environmental	0.00638	equipment	0.01101	models	0.00891	magnetic	0.00736
understanding	0.00573	studies	0.00604	project	0.01023	analysis	0.00799	electron	0.0068
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
research	0.04002	molecular	0.01428	data	0.01493	students	0.04399	system	0.01947
scientists	0.01163	cell	0.01152	water	0.01087	science	0.03092	design	0.01944
national	0.01114	protein	0.01101	processes	0.00908	project	0.02327	systems	0.01931
researchers	0.01018	specific	0.01058	study	0.00893	program	0.01754	control	0.01329
workshop	0.00976	function	0.00979	model	0.00772	engineering	0.01521	based	0.01257
scientific	0.00958	cells	0.00942	flow	0.00761	education	0.01385	performance	0.01067
field	0.00911	studies	0.00894	ocean	0.00748	undergraduate	0.0127	data	0.01043
support	0.009	proteins	0.0089	climate	0.0067	laboratory	0.01098	develop	0.00939
areas	0.00894	mechanisms	0.00883	ice	0.00629	faculty	0.01078	network	0.00839
international	0.00886	dna	0.00804	field	0.00622	learning	0.0107	software	0.00814

TOPIC 1		TOPIC 2		TOPIC 3		TOPIC 4		TOPIC 5	
research	0.7831	theory	0.3742	materials	0.465	species	0.392	mathematics	0.4311
project	0.2861	analysis	0.2975	engineering	0.3472	molecular	0.3605	education	0.3067
students	0.228	model	0.2848	chemistry	0.3194	dr	0.3161	teachers	0.2793
university	0.2101	models	0.2846	design	0.309	chemical	0.3081	physics	0.2777
program	0.1897	problems	0.2702	laboratory	0.2879	surface	0.2715	year	0.2727
science	0.1515	understanding	0.2536	computer	0.2436	experiments	0.2677	school	0.2646
data	0.1466	studies	0.2491	state	0.2061	phase	0.2373	faculty	0.2498
study	0.1448	methods	0.2455	technology	0.2044	process	0.2359	undergraduate	0.2371
support	0.1356	important	0.2374	techniques	0.2022	determine	0.2258	college	0.2279
systems	0.1292	information	0.2273	properties	0.1972	effects	0.2014	student	0.2183
TOPIC 6		TOPIC 7		TOPIC 8		TOPIC 8		TOPIC 10	
cell	0.4392	abstract	0.7037	group	0.4609	order	0.4939	equipment	0.5951
protein	0.3731	fellowship	0.51	groups	0.4016	experimental	0.2891	nsf	0.362
cells	0.3254	postdoctoral	0.3304	areas	0.3214	theoretical	0.2804	projects	0.3613
proteins	0.2873	required	0.2857	number	0.2699	dynamics	0.2796	grant	0.2695
plant	0.2709	error	0.113	area	0.2677	scientific	0.2789	network	0.2596
gene	0.2686	mathematical	0.1006	water	0.231	task	0.2612	funds	0.1912
genes	0.2525	sciences	0.0961	focus	0.2276	test	0.243	performance	0.1826
dna	0.2093	worry	0.0846	behavior	0.2159	scientists	0.2212	community	0.1692
function	0.2075	matter	0.0462	environmental	0.2149	level	0.2193	instrumentation	0.1597
biology	0.1953	length	0.044	related	0.2063	national	0.215	magnetic	0.15

Table 13: Comparison between LDA (top) and Sparse PCA (bottom) on the NSF data set.

Topic	LDA	SPCA
1	Project	Research
2	Species	Theory
3	Research	Materials
4	Theory	Species
5	Materials	Mathematics/Education
6	Research	Cell
7	Molecular	Abstract
8	Data/Climate	Research Group
9	Students	Order
10	Systems	Equipment

Table 14: Manually associated topics in the NSF data set experiment.

Data set	Amazon	Reuters	NSF
LDA	1 minute	13 minutes	1 hour
SPCA	19 seconds	3 minutes	14 minutes

Table 15: Computational times for LDA and sparse PCA.

The two methods share many topics in common. LDA's topic 9 and SPCA's topic 1 are both on university education, with terms such as "students" and "undergraduate". LDA's topic 5 and SPCA's topic 3 are both related to material science and physics. LDA's topic 7 and SPCA's topic 6 focus on molecular and cell biology, protein function and gene expression. Overall the LDA method appears to behave slightly better on this data set; sparse PCA provides a few topics (8, 9) without clear and consistent meaning. LDA does provide topics with overlap, and/or without much specificity (topics 1 and 3 for example), while non-overlap is automatically enforced with sparse PCA. As discussed next, sparse PCA runs much faster than LDA.

4.4 Comparison summary

To summarize our findings, we note that overall both LDA and sparse PCA behave well and comparably on the data sets we have used. In the larger data set (NSF) LDA delivers better results. A clear advantage, besides performance, of sparse methods lies with their ease of use and readability; our matlab code for sparse power iteration is a few lines long, and is quite amenable to a distributed computing architecture. Another clear advantage lies with the computational effort that is required. To our knowledge, there is no precise computational complexity analysis for both methods. In our experiments we have observed that LDA takes much longer to run than sparse PCA. Table 15 illustrates the dramatic difference in run times[4].

5 Conclusions and future work

Sparse learning problems are formulated as optimization problem with explicit encoding of sparsity requirements, either in the form of constraint or via a penalty on the model variables. This encoding leads to a higher degree of interpretability of the model without penalizing, in fact *improving*, the computational complexity of the algorithm. As such, the results offer an explicit trade-off between accuracy and sparsity, based on the value of the sparsity-controlling parameter that is chosen. In comparison to thresholded PCA, LDA or similar methods, which provide "after-the-fact" sparsity, sparse learning methods offer a principled way to explicitly encode the trade-off in the optimization problem. Thus, the enhanced interpretability of the results is a direct result of the optimization process.

We demonstrated the sparse learning techniques on a real-world data set from the Aviation Safety Reporting System and showed that they can reveal contributing factors to aviation safety incidents such as runway incursions. We also show that the sparse PCA and LASSO algorithms can discover the underlying task hierarchy that pilots perform. We have compared the LDA and sparse PCA approaches on other commonly used data sets. Our numerical experiments indicate that the sparse PCA and LASSO methods are very competitive with respect to thresholded methods (involving say LDA and naïve Bayes), at very moderate computational cost.

In the safety monitoring of most critical, large-scale complex systems, from flight safety to nuclear plants, experts have relied heavily on physical sensors and indicators (temperature, pressure, etc). In the future we expect that human-generated text reporting, assisted by automated text understanding tools, will play an ever increasing role in the management of critical business, industry or government operations. Sparse modeling, by offering a great trade-off between user interpretability and computational scalability, appears to be well equipped to address some of the corresponding challenges.

[4]We have used a 64-bit 2.33 GHz quad core Dell precision 690 desktop running Red Hat Enterprise Linux (version 5.4) having 24GB of physical memory. The LDA code is a Matlab compiled mex routine run on Matlab version 2011; the sparse PCA code is a few lines of ordinary matlab.

6 Acknowledgments

A.N.S. thanks the NASA Aviation Safety Program, System Wide Safety and Assurance Technologies project for supporting this work, and Dr. Irving Statler, Linda Connell, and Charles Drew for their valuable insights regarding the Aviation Safety Reporting System and related discussions. L.E.G.'s work is partially supported by the National Science Foundation under Grants No. CMMI-0969923 and SES-0835531, as well as by a University of California CITRIS seed grant, and a NASA grant No. NAS2-03144. The authors would like to thank the Associate Editor and the reviewers for their careful review of this manuscript.

A ASRS Data Preparation

In this section, we elucidate our data preparation processes by first showing a sample ASRS report:

> WHILE taxi TO runway OUT OF GATE AT sdf airport I unintentional taxi ON runway.WHEN I notice I WAS ON THE runway I HAD MY firstofficer TELL THE tower WE WERE ON THE activerunway.THIS happen WHILE THE firstofficer WAS call OUT THE takeoff VREF number.I WAS look IN AND OUT OF THE COCKPIT.THIS IS WHEN I SAW THE RED runway mark.I DID NOT NOTICE THE ARROW NEXT TO runway.THE ARROW point TO THE taxiway I TOOK taxiway G.THIS WAS WHAT disorient ME.THE problem WAS THAT THE SIGN WAS NOT PARALLEL WITH THE runway WHICH IS WHAT I AM normal us TO.THE SIGN WAS MORE PARALLEL TO taxiway G.WHAT SHOULD HAVE BEEN A SIMPLE TAXI TO THE END OF THE runway SOMEHOW turn INTO A runwayincursion.I THINK THE SIGN SHOULD BE TO THE left south OF taxiway G AND PARALLEL TO runway.

Here is the pre-processing sequence:

1. We revert all characters to lower case and scrub all punctuation and special characters. We remove redundant white spaces. We remove stop words[5].

2. We glue each runway or taxiway and their labels as a single word. This is achieved by the following regular expression:

   ```
   /\b(runway|taxiway)\ [a-z0-9]{1,3}\b/
   ```

   This converts `taxiway xy` to `taxiway_xy`, or `runway 9x` to `runway_9x`.

   The purpose of this step is to enable us to just focus on the issues raised by the runways, by the taxiways, or anything else. Additionally, this allows for immediate perception of machine learning results by the reviewer. For example, we would rather expect a list of other words associated to an airport as `taxiway_9l`, `runway_00`, `runway_y` than simply `9l`, `00`, `y` etc.

3. We tokenize each ASRS report by separating the words of the ASRS report using a single space.

4. We vectorize the text by extracting all uni-grams from the sample. In doing so, we first scan through all ASRS documents and build a dictionary of all uni-grams ever used across a total of $21,519$ ASRS reports. We thus find $27,081$ distinct uni-grams. Hence, we obtain a data matrix of dimension $21,519 \times 27,081$.

With this matrix, we have run LASSO, sparse PCA, and LDA as described above.

[5]Using the list at `http://www.eecs.berkeley.edu/~gawalt/MIR2010/NYTWStops.txt`.

References

[1] M. A. U. Abedin, V. Ng, and L. Khan. Cause identification from aviation safety incident reports via weakly supervised semantic lexicon construction. *J. Artif. Int. Res.*, 38:569–631, May 2010.

[2] A. Agovic and A. Banerjee. Gaussian process topic models. In *Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 10–19, Corvallis, Oregon, 2010.

[3] M. S. Ahmed and L. Khan. SISC: A Text Classification Approach Using Semi Supervised Subspace Clustering. In *Proceedings of the 2009 IEEE International Conference on Data Mining Workshops*, ICDMW '09, pages 1–6, 2009.

[4] A. Amini and M. Wainwright. High-dimensional analysis of semidefinite relaxations for sparse principal components. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 2454–2458. IEEE, 2008.

[5] J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *The Journal of Machine Learning Research*, 3:1229–1243, 2003.

[6] D. Blei and J. McAuliffe. Supervised topic models. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 121–128. MIT Press, Cambridge, MA, 2008.

[7] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[8] P. Bühlmann and B. Yu. Sparse boosting. *The Journal of Machine Learning Research*, 7:1001–1024, 2006.

[9] E. Candès and Y. Plan. Near-ideal model selection by ℓ_1 minimization. *Annals of Statistics*, 37:2145–2177, 2009.

[10] D. Das and A. F. T. Martins. A survey on automatic text summarization, 2007.

[11] A. d'Aspremont, F. Bach, and L. El Ghaoui. Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research*, 9:1269–1294, 2008.

[12] B. Ding, D. Lo, J. Han, and S.-C. Khoo. Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, pages 1024–1035, 2009.

[13] B. Ding, B. Zhao, C. X. Lin, J. Han, and C. Zhai. Topcells: Keyword-based search of top-k aggregated documents in text cube. *Data Engineering, International Conference on*, pages 381–384, 2010.

[14] A. Dobra, C. Hans, B. Jones, J. R. Nevins, G. Yao, and M. West. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196 – 212, 2004. Special Issue on Multivariate Methods in Genomic Data Analysis.

[15] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[16] J. Eisenstein, A. Ahmed, and E. P. Xing. parse additive generative models of text. In *International Conference on Machine Learning (ICML)*, 2011.

[17] L. El Ghaoui, G.-C. Li, V.-A. Duong, V. Pham, A. Srivastava, and K. Bhaduri. Sparse machine learning methods for understanding large text corpora. In *Conference on Intelligent Data Understanding*, Oct. 2011.

[18] L. El Ghaoui, V. Viallon, and T. Rabbani. Safe feature elimination for the LASSO. To appear in *Pacific Journal of Optimization*, Special Issue on Conic Optimization, Nov. 2012.

[19] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[20] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[21] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

[22] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432, 2008.

[23] B. Gawalt, J. Jia, L. Miratrix, L. El Ghaoui, B. Yu, and S. Clavier. Discovering word associations in news media via feature selection and sparse classification. In *Proc. 11th ACM SIGMM International Conference on Multimedia Information Retrieval*, 2010.

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

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

[26] T. Joachims. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer/Springer, 2002.

[27] J. E. Jonsson and W. R. Ricks. Cognitive models of pilot categorization and prioritization of flight-deck information. Technical report, 1995.

[28] M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *arXiv:0811.4724*, 2008.

[29] M. Kolar, A. Parikh, and E. Xing. On Sparse Nonparametric Conditional Covariance Selection. *International Conference on Machine Learning*, 2010.

[30] C. X. Lin, B. Ding, J. Han, F. Zhu, and B. Zhao. Text Cube: Computing IR Measures for Multidimensional Text Database Analysis. *IEEE International Conference on Data Mining*, pages 905–910, 2008.

[31] Z. Lu, R. Monteiro, and M. Yuan. Convex optimization methods for dimension reduction and coefficient estimation in multivariate linear regression. *Mathematical Programming*, 9(1):1–32, 2010.

[32] L. Mackey. Deflation methods for sparse PCA. *Advances in Neural Information Processing Systems*, 21:1017–1024, 2009.

[33] J. Mairal, G. Sapiro, and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modeling and Simulation*, 7(1):214–241, 2008.

[34] N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37(1):246–270, 2008.

[35] B. Moghaddam, Y. Weiss, and S. Avidan. Fast Pixel/Part Selection with Sparse Eigenvectors. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

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

[37] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *CORE Discussion Papers*, 2010.

[38] O.Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, March 2008.

[39] N. C. Oza, J. P. Castle, and J. Stutz. Classification of Aeronautics System Health and Safety Documents. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 39(6):670–680, 2009.

[40] I. Persing and V. Ng. Semi-supervised cause identification from aviation safety reports. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, pages 843–851, 2009.

[41] F. Schilder and R. Kondadadi. Fastsum: fast and accurate query-based multi-document summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 205–208, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[42] H. Shan, A. Banerjee, and N. C. Oza. Discriminative Mixed-Membership Models. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pages 466–475, Washington, DC, USA, 2009.

[43] H. Shen and J. Z. Huang. Sparse principal component analysis via regularized low rank matrix approximation. *J. Multivar. Anal.*, 99:1015–1034, July 2008.

[44] J. Songsiri and L. Vandenberghe. Topology selection in graphical models of autoregressive processes. *Journal of Machine Learning Research*, 2010.

[45] M. Steyvers and T. Griffiths. Matlab topic modeling toolbox version 1.4, 2011.

[46] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal statistical society, series B*, 58(1):267–288, 1996.

[47] J. A. Tropp. Just relax: Convex programming methods for identifying sparse signals. *IEEE Trans. Inform. Theory*, 51(3):1030–1051, Mar. 2006.

[48] C. Woolam and L. Khan. Multi-concept Document Classification Using a Perceptron-Like Algorithm. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, pages 570–574, 2008.

[49] C. Woolam and L. Khan. Multi-label large margin hierarchical perceptron. *IJDMMM*, 1(1):5–22, 2008.

[50] M. Yuan and Y. Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19, 2007.

[51] D. Zhang, C. Zhai, J. Han, A. Srivastava, and N. Oza. Topic modeling for OLAP on multidimensional text databases: topic cube and its applications. *Stat. Anal. Data Min.*, 2:378–395, December 2009.

[52] Y. Zhang, A. d'Aspremont, and L. El Ghaoui. Sparse PCA: Convex relaxations, algorithms and applications. In M. Anjos and J. Lasserre, editors, *Handbook on Semidefinite, Cone and Polynomial Optimization: Theory, Algorithms, Software and Applications*, International Series in Operational Research and Management Science. Springer, 2012.

[53] Y. Zhang and L. El Ghaoui. Large-scale sparse principal component analysis and application to text data. In *Proc.* Conf. Neural Information and Processing Systems, Dec. 2011.

[54] P. Zhao and B. Yu. Stagewise lasso. *The Journal of Machine Learning Research*, 8:2701–2726, 2007.

[55] H. Zou, T. Hastie, and R. Tibshirani. Sparse Principal Component Analysis. *Journal of Computational & Graphical Statistics*, 15(2):265–286, 2006.