Classification of Aeronautics System Health and Safety Documents

Nikunj Oza, *Member, IEEE,* J. Patrick Castle, *Member, IEEE,* and John Stutz, *Member, IEEE,*

Abstract—**Most complex aerospace systems involve large numbers of text reports relating to safety, maintenance, and associated issues. The Aviation Safety Reporting System (ASRS) database spans several decades and contains over 700,000 reports. The Aviation Safety Action Plan (ASAP) contains more than 12000 reports from various airlines. Problem categorizations have been developed for both ASRS and ASAP to enable identification of key system problems. However, repository volume and complexity can make human analysis difficult. Multiple human experts are needed, and they often disagree on their classifications. Consistent classification is necessary to support tracking trends in problem categories over time. A decision support system that performs consistent document classification quickly and over large repositories would be very useful.**

We discuss the results of two algorithms we have developed and implemented to classify ASRS and ASAP documents. The first is Mariana—a Support Vector Machine with Simulated Annealing, which is used to find the best hyperparameters for the model. The second method is classification built on top of Nonnegative Matrix Factorization (NMF), which attempts to find a document model that represents document features that add up in various combinations to form documents. We tested both methods on ASRS documents and ASAP documents with the latter categorized two different ways. We illustrate the potential of NMF to provide document features that are interpretable and indicative of topics. We also briefly discuss the tool that we have incorporated Mariana into in order to allow human experts to provide feedback on the document categorizations.

I. Introduction

Most complex aerospace systems have large numbers of reports relating to requirements, design, manufacturing, operation, maintenance, safety, and/or associated issues. Such systems often also have a formal problem reporting system. Problem reporting systems can be quite large, containing thousands of reports, spanning many years, and could even bridge several databases. The Aviation Safety Reporting System (ASRS), despite its constrained scope, spans several decades and contains over 700,000 reports [1]. The Aviation Safety Action Plan (ASAP) is a system that many airlines have chosen to implement and the combined archive contains more than 12000 reports. Both repositories are currently growing. These numbers of reports are large enough that relying entirely on humans to extract knowledge from these reports would be impractical.

These information repositories contain much valuable information about many aspects of aircraft in the NAS. For these repositories, problem categorizations have been developed to enable identification of groups of related reports and analysis of trends in the rates of occurrence of different problems. Our primary concern is system health and safety—particularly detection, diagnosis, prediction, mitigation, and prevention of ongoing and future system problems. However, repository volume and complexity make human analysis difficult. The large number of reports mandates dividing them among multiple human experts to analyze and categorize the reports. However, experts are known to disagree on the most suitable anomaly categories to use, especially in systems such as ASRS and ASAP where the categories are not mutually exclusive. Also, we have observed that the same expert analyzing the same report at different times arrives at different problem categorizations. For anomaly detection and trend analysis on large datasets or across multiple datasets, consistent categorization is necessary. A decision support system that automatically analyzes reports and provides consistent classifications would be extremely useful.

In this paper, we mainly discuss the results of two classification algorithms that we have developed and implemented to classify ASRS and ASAP documents. The first is Mariana, which is a Support Vector Machine with Simulated Annealing, which is used to find the best hyperparameters for the model. The second method is classification built on top of Nonnegative Matrix Factorization (NMF), which attempts to find a document model that represents document features that add up in various combinations to form documents. We tested both methods on ASRS documents and ASAP documents with the latter categorized two ways. One categorization is referred to as event types, while the second categorization uses ASAP contributing factors, which is a categorization intended to represent causal factors for the ASAP event types. On the ASRS data, we illustrate the potential of NMF-based classification to provide basis vectors that are interpretable and indicative of different topics present within the repository. We also briefly discuss the tool that we have incorporated Mariana into in order to allow human experts to provide feedback on the document categorizations. We plan to incorporate NMF-based classification and other methods developed in the future into this tool.

In the next section, we survey some related work on text classification and the choices that have to be made in implementation. In section 3, we discuss the datasets on which we tested our methods. In section 4, we describe Mariana, while in section 5, we describe the Nonnegative Matrix Factorization (NMF)-based classifier that we used. In section 6, we compare our algorithms' performances on the datasets. In Section 7 we briefly describe the tool developed to display aviation safety

Nikunj Oza and John Stutz are with NASA Ames Research Center, Moffett Field, CA 94035-1000, USA.

J. Patrick Castle is with Mission Control Technologies, Moffett FIeld, CA 94035-1000, USA.

documents and Mariana's classifications of those documents, as well as allow human feedback on those classifications. In Section 7 we describe our plans for further developments and in section 8 we conclude this paper.

II. Related Work and Research

In this section, we briefly describe related work in text classification. A more comprehensive survey of text classification and text mining in general can be found in [20]. In the area of text classification, there are two key decisions that have to be made: representation of the text and the classification method used. The documents in their raw form are not amenable to machine learning, so another representation is needed. The simplest document representation is the bag of words (BOW). In this representation, each document is represented by a vector consisting of as many elements as there are unique words in the document repository. Each vector element represents the number of times that a given word occurs in that document. The repository can be represented by a (BOW) matrix in which each document's vector is a row in the matrix. In machine learning terminology, each document is an example or instance and each word frequency is a feature. In principle, one could then apply one of many possible classification algorithms to document repositories after converting them to BOW matrices. One difficulty with this scheme is the weighting of words. Both rare words and very common words should be given low weight in classification because both types of words are not useful for classification, where one generally wants features that come as close as possible to splitting the set of examples into equal-sized bins. However, very common words are given very high weight due to their high word frequency. For this reason, weighting schemes such as term frequency inverse document frequency (TFIDF) are often used. TFIDF multiplies the term frequency by the inverse document frequency, which is related to the reciprocal of the fraction of documents in which the word appears. This reduces the weights of very common words. Term frequency and even TFIDF give excessive weight to long documents, so these are often normalized so that all documents' vectors have equal length. Stop words, which are very common words such as articles, are often eliminated from documents using a stopword list.

The bag of words and its weighted variants have one major weakness—they lose all semantic information because the order of words is lost. A given document is treated exactly the same way no matter how its words are reordered. Semantic information is clearly very important in human understanding of text. Natural Language Processing (NLP) methods attempt to maintain semantic information on documents while making the representation more amenable to machine learning. One example is a system called PLADS, which expands acronyms and collapses phrases so that different expressions with exactly the same meaning (e.g., FL vs. flight level) are reduced to one word or phrase. PLADS also sets nouns to their singular form and verbs to the infinitive conjugation. Others have attempted to use pairs or triples of words as features rather than individual words. Full parse trees can be constructed in which the subject and object for each sentence are identified and sometimes broken down further into nouns, adjectives, prepositional phrases, etc. The chief drawback of such methods is that they are typically computationally very intensive. Also, it appears that the collocation of terms, which is maintained in the bag of words representation, is often sufficient for classification, because NLP methods have so far yielded too little improvement in classification performance to justify their computation time. We demonstrate this partially in this paper.

Many of the same methods used for non-text classification, such as decision trees, random forests, and Support Vector Machines, are used for text [10], [20]. Past experiments have shown that Support Vector Machines seem to be especially suitable because they are naturally suited to dealing with very large numbers of features, which typically exist in text classification problems [20]. We demonstrate that in this paper.

III. Datasets

In this section, we describe the three datasets that we use for our experiments: ASRS, ASAP with event types, and ASAP with contributing factors.

A. ASRS Documents

The Aviation Safety Reporting System (ASRS), created 30 years ago, is a joint effort by the FAA and NASA [1]. The program was established to collect data and information about aviation events which could lead to unsafe situations, or non-standard procedures, and use the data and information to identify deficiencies and discrepancies in the NAS so appropriate solutions could be implemented. ASRS reports are publicly available and are written by pilots, flight controllers, technicians, flight attendants, and others including passengers. In general, the reports are filed in response to a specific event, but general concerns and complaints are also filed. ASRS reports include factual information about the aircraft, location, parties involved and a narrative in which the author describes the event(s) and/or situation. Since these narratives are specific to aviation they are filled with acronyms and abbreviations that only those with a working knowledge of the industry are likely to understand. Each report is read by at least two aviation experts who identify hazardous conditions and underlying causes of the reported events. There are now over 700,000 documents in the ASRS data base with more than 3000 reports added each month. Many air carriers also have their own internal safety reporting system under the Aviation Safety Action Plan (ASAP) which may or may not be linked to the ASRS. Different airlines devised different categorizations for their documents, which led to the development of a master version of the categorizations which all organizations could reference. This version is known as the Distributed National Action Archive (DNAA).

Aviation-related documents are quite different from existing data sets, such as Reuters-21578 and 20 Newsgroups [21]. Industry specific terms, references to equipment and positions, and the use of acronyms differentiate the aviation-related reports from text that the public is commonly exposed to. The intended audience of aviation reports is expected to have a working knowledge of the industry so the reports are often void of descriptive or background material. The Reuters and 20 Newsgroups data sets are much more verbose and use much more everyday common language.

For our research and development of text categorization algorithms for the aeronautics domain we have prepared subsets of ASRS and ASAP reports. Since the Aviation Safety Reporting System (ASRS) narratives are publicly available, we identified a subset of the ASRS reports which are representative of the entire data set and used this subset for our analysis. Many categories in the original ASRS have very few associated reports, which makes assessment of classification algorithms difficult. For this reason, the subset of documents we used only contains 22 categories from the original 60, and contains 28596 documents.

TABLE I
DISTRIBUTION OF THE 28596 SELECTED ASRS DOCUMENTS OVER THE 22 DNAA CATEGORIES.

Cat#	#Docs	%Docs	-	Cat#	#Docs	%Docs
1	1876	6.56		12	4275	14.95
2	16173	56.56		13	2849	9.96
3	615	2.15		14	1654	5.78
4	610	2.13		15	508	1.78
5	3915	13.69		16	1249	4.37
6	7663	26.80		17	556	1.94
7	2230	7.80		18	1490	5.21
8	2844	9.95		19	8534	29.84
9	573	2.00		20	876	3.06
10	1456	5.09		21	441	1.54
11	514	1.80		22	807	2.82
				total	61708	215.78

Table I indicates both the non-exclusive nature of the subject matter expert's assigned categories and the great imbalance between category populations, with ratios up to 37:1. Class 2 is assigned to more than half the narratives, classes 6 and 19 to more than a quarter each, while 12 of the 22 have populations under 6% of the total.

Table II highlights the exceedingly non-exclusive distribution of the subject matter expert's assigned DNAA categories. On average, the narratives are assigned to 2.16 categories, with one document assigned to 10 categories. Less than a third are assigned to single categories, more than a third have 2 categories, while the remaining third have 3 or more categories.

Table III describes the term distributions over the documents by filtering the terms with respect to the number of documents they occur in. The column labeled 'minD' gives the minimum number of documents per term. The next two columns give the number of terms and percentage of total terms that appear in at least the corresponding number of documents appearing in the first column. Thus row 1 includes the entire stop word filtered set of 25729 terms, which is 100% of the full term set, since they all appear in at least one document. The column labeled 'Nonzero' gives the number of distinct term occurrences, which is the number of non-zero entries in the BOW matrix after choosing only the columns corresponding to the corresponding number of terms. The 'Sparsity' column

TABLE II
NUMBER OF CATEGORIES PER DOCUMENT WITHIN THE SELECTED ASRS DOCUMENT SET (28596 DOCUMENTS AND 22 CATEGORIES).

#Cats	#Docs	%Docs
1	9032	31.585
2	9926	34.711
3	6915	24.182
4	1911	6.683
5	565	1.976
6	154	0.539
7	68	0.238
8	16	0.056
9	8	0.028
10	1	0.004

TABLE III
DISTRIBUTION OF STOPWORD FILTERED 2+CHARACTER TERMS WITHIN SELECTED ASRS DOCUMENT SET.

minD	#Term	%Term	Nonzero	Sparsity	Totals	%Total
1	25729	100.00	1714931	0.23	2615965	100.00
2	17603	68.42	1706805	0.34	2605072	99.58
4	12560	48.82	1694903	0.47	2588504	98.95
8	8991	34.94	1676237	0.65	2563263	97.98
16	6103	23.72	1644694	0.94	2521891	96.40
32	4065	15.80	1599497	1.38	2464005	94.19
64	2685	10.44	1537551	2.00	2384040	91.13
128	1731	6.73	1452340	2.93	2273663	86.91
256	1068	4.15	1331842	4.36	2113264	80.78
512	609	2.37	1164957	6.69	1891332	72.30
1024	351	1.36	978935	9.75	1638920	62.65
2048	162	0.63	707972	15.28	1267763	48.46
4096	69	0.27	445444	22.58	868293	33.19
8192	14	0.05	151219	37.77	364014	13.92
16384	1	0.00	17070	59.69	51928	1.98

is the corresponding matrix sparsity—the percentage of entries in the bag of words matrix that are nonzero. The column 'Totals' is the total number of subset term occurrences, while '%Total' is the percentage of total term occurrences in the repository that represent subset term occurrences. Returning to row 1, the 25729 terms that occur in at least one document appear 1714931 times in the repository if a term's appearance multiple times within a document is only counted as one appearance, and appear 2615965 total times in the repository when counting multiple appearances within documents.

There are several interesting statistics here. Since 100% of the terms are found in at least one document while 68.42% of documents are found in at least two documents, we have that about 32% of documents appear in only one document. These are useless for text classification. Similarly, terms that are found in a very large numbers of documents are useless for classification—0.05% of the terms are found in at least 8192 (28%) of the documents. Unless their distributions have strong statistical properties, these terms also provide little information, and so become candidates for an ASRS domain specific stopword list.

Figure 1 illustrates the 22 DNAA categories' binary population correlations, i.e., the relative degree of co-occurrence. The strongest correlations, which are still quite weak, involve the three largest population categories. The largest correlations involve categories 2, 6, and 10 with correlations ranging from

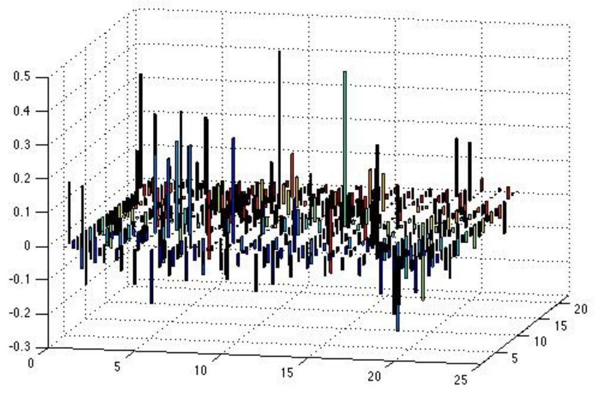


Fig. 1. Binary correlations for the ASRS categories. Diagonal elements, all with value 1.0, have been suppressed so that the other correlations are more clearly visible.

0.47 to 0.33. There are also a fair number of 3-way correlations, but the number of significant higher level correlations decreases rapidly. High correlations between large and small population categories would imply that essentially all instances assigned to the smaller category are also assigned to the larger.

Since there are no obviously strong correlations between the different categories, we assume category independence. So, instead of building one large multi-category classifier we treated each category individually and set up a series of binary (e.g., 'in the class' or 'out of the class') classification problems. This gave us some extra flexibility, but by treating each report as being either "in" or "out" of a category we added to the imbalance problem of the data by increasing the number of reports considered "out" of each category.

All of these factors work against the direct applicability of most conventional statistical text mining techniques, which have largely been developed for data sets having significantly different statistical properties. A significant motivation for this work was finding text classification methods that would work on our repositories even though they are very challenging for text classification.

B. ASAP Documents

The ASAP narratives proprietary; however, we can report some general statistics and the results of our algorithms on these documents.

Table IV shows that the ASAP event types are not as imbalanced as the ASRS document set, since no more than 21% of the documents fall into any one ASAP event type whereas nearly 57% of the documents fall into one ASRS category. The ASAP event types are also closer to being mutually exclusive than the ASRS categories, since the "total" line of the table indicates that each document is assigned to about 1.09 categories on average, compared to nearly 2.16 for ASRS.

Table V corroborates the greater tendency of ASAP documents to be assigned to only one category. About 60% of the ASAP documents are assigned to only one event type, whereas only about 32% of ASRS documents are assigned to one category. Nearly 18% of the documents are not classified at all. These documents may still be useful for some algorithms such as our NMF-based classification algorithm.

Table VI shows that the terms in the ASAP documents vary wildly in terms of their occurrences within the documents just like ASRS. About 37% of the terms are found in three documents or less. Therefore, these are not very useful for statistical text mining. The ASAP documents tend to have a somewhat smaller percentage of terms appearing in a large number of documents compared to ASAP.

Figure 2 illustrates the correlations between the ASAP event types. The correlations are generally low just as with ASRS; therefore, we used the same one-vs-all classification methodology as with ASRS.

C. ASAP with Contributing Factors

Table VII shows that the contributing factors are in between the ASAP event types and ASRS categories in terms of

TABLE IV
DISTRIBUTION OF THE 11245 ASAP DOCUMENTS, OVER THE 33 ASAP CATEGORIES.

Cat#	#Docs	%Docs	-	Cat#	#Docs	%Docs
1	2362	21.00		12	1880	16.72
2	68	0.60		13	1001	8.90
3	1317	11.71		14	540	4.80
4	1449	12.89		15	19	0.17
5	182	1.62		16	71	0.63
6	78	0.69		17	420	3.73
7	86	0.76		18	270	2.40
8	74	0.66		19	86	0.76
9	6	0.05		20	244	2.17
10	80	0.71		21	269	2.39
11	47	0.42		22	83	0.74

Cat#	#Docs	%Docs
23	134	1.19
24	260	2.31
25	215	1.91
26	469	4.17
27	293	2.61
28	14	0.12
29	0	0
30	29	0.26
31	6	0.05
32	127	1.13
33	25	0.22
total	12204	108.53

TABLE V
DISTRIBUTION OF 11245 ASAP DOCUMENTS AND 33 EVENT TYPES.

#Cats	#Docs	%Docs
0	2004	17.82
1	6775	60.25
2	2065	18.36
3	325	2.89
4	61	0.54
5	12	0.11
6	1	0.009
7	2	0.018

TABLE VI
DISTRIBUTION OF WORDS WITHIN ASAP DOCUMENTS.

minD	#Term	%Term	nnz	%SP	totals	%total
1	14686	100.00	722488	0.44	1211872	100.00
2	14686	100.00	722488	0.44	1211872	100.00
4	9242	62.93	709800	0.68	1193767	98.51
8	6215	42.32	694283	0.99	1171274	96.65
16	4195	28.56	672448	1.42	1139192	94.00
32	2769	18.85	641081	2.06	1094147	90.29
64	1789	12.18	597699	2.97	1030608	85.04
128	1118	7.61	537450	4.28	942033	77.73
256	641	4.36	453441	6.29	816939	67.41
512	298	2.03	332617	9.93	631194	52.08
1024	103	0.70	195104	16.84	395805	32.66
2048	28	0.19	90301	28.68	190957	15.76
4096	5	0.03	29789	52.98	55331	4.57
8192	1	0.01	9326	82.93	11527	0.95
16384	0	0	0	—	0	0.00

TABLE VII
NON-EXCLUSIVE CATEGORY POPULATIONS

Distribution of the 11245 ASAP documents, w.r.t. the 24 ASAP contributing factors.

Cat#	#Docs	%Docs	-	Cat#	#Docs	%Docs
1	2252	20.03		13	251	2.23
2	205	1.82		14	799	7.11
3	199	1.77		15	23	0.20
4	276	2.45		16	466	4.14
5	141	1.25		17	1074	9.55
6	1400	12.45		18	699	6.22
7	331	2.94		19	1084	9.64
8	290	2.58		20	218	1.94
9	361	3.21		21	825	7.34
10	1050	9.34		22	378	3.36
11	1062	9.44		23	834	7.42
12	35	0.31		24	1081	9.61
				total	15334	136.36

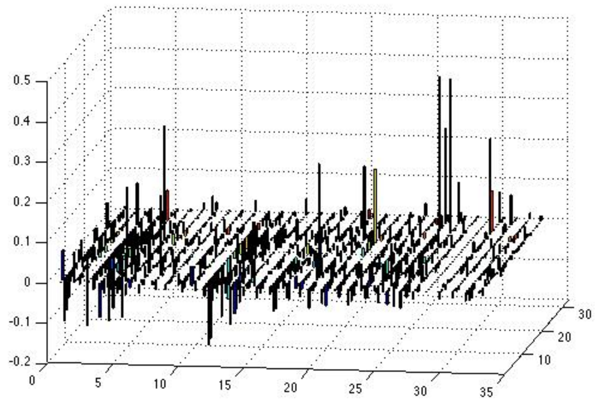


Fig. 2. Binary correlations of ASAP event types. Diagonal elements, all with value 1.0, have been suppressed.

TABLE VIII
NON-EXCLUSIVE CATEGORIZATION, DISTRIBUTION OF 11245 DOCUMENTS AND 25 CONTRIBUTING FACTORS.

#Cats	#Docs	%Docs
0	3317	29.50
1	4192	37.28
2	1974	17.55
3	831	7.39
4	427	3.80
5	255	2.27
6	125	1.11
7	62	0.55
8	37	0.33
9	19	0.17
10	2	0.02
11	1	0.01
12	3	0.03

the amount of overlap in the categories, as shown by the 136.36% of total documents covered, which indicates that each document has an average of 1.36 assigned categories. The contributing factors are also better balanced—only 2 of the 24 contributing factors have fewer than 1% of the documents, whereas nearly 16 out of the 33 event types have fewer than 1% of the documents.

Table VIII highlights the non-exclusive distribution of the subject matter expert's assigned DNAA contributing factors. Documents are assigned to a maximum of 12 contributing factors. Only about 37% are assigned to single categories, and about 30% are not assigned to any contributing factors at all.

Figure 3 illustrates the correlations between the ASAP contributing factors. The correlations are generally low just as with the ASAP event types with a few exceptions; therefore, we assume category independence and use the same one-vs-all classification methodology for ASAP contributing factors as for the other categorizations.

Figure 4 illustrates the correlations between the ASAP event types and contributing factors. The correlations are generally quite low. The relationships between the contributing factors and event types may be more complicated and may require other information about the flights described in the reports. Exploring this possibility will be the subject of future work.

IV. SUPERVISED LEARNING

Supervised learning takes data for which both inputs and outputs are provided as a training set and uses them to build

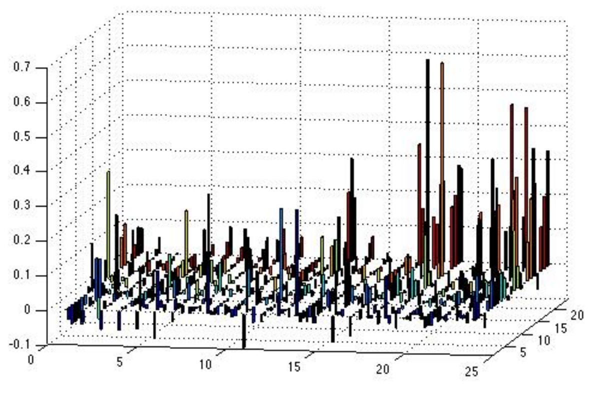


Fig. 3. Binary correlations of ASAP contributing factors

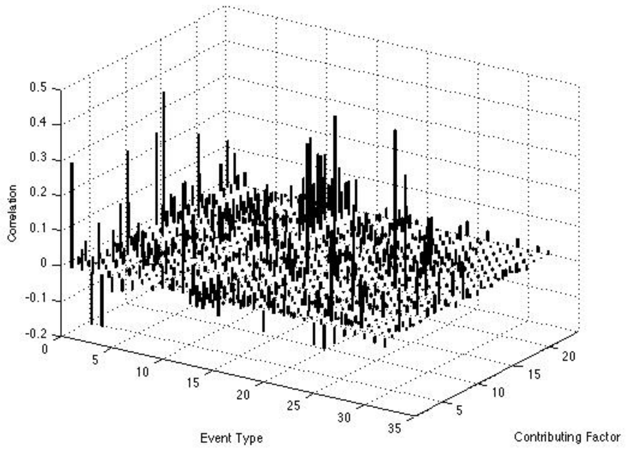


Fig. 4. Binary correlations of ASAP event types vs. contributing factors.

a model intended to represent the mapping from inputs to outputs. This model is then used to generate predicted outputs for new data for which only inputs have been provided. In our problem, the training set consists of documents that have already been assigned categories. The model generated by a supervised learning method is then used to categorize reports that have previously not been classified. There are many different supervised learning methods, each of which has its own strengths and weaknesses. We set out to build a generic tool with algorithms that have not been inherently designed to excel in the aviation domain. The point is to avoid the time and expense associated with building a system specific to ASRS or ASAP and see how well the system would perform without such engineering. Additionally, such effort may lead to overfitting our data, just as excessively complex machine learning models often overfit the training data.

A disadvantage of supervised learning techniques is that they are only as good as the training data. Though the aviation report reviewers who set the labels for the training data are experts and work hard to be consistent, they are human, and they frequently disagree on the proper classification(s) of a report. Even a particular expert who examines the same report at different times may classify it differently each time. Therefore, one may argue that using automated classification will not improve results over humans and that our models will not represent "ground truth," but rather the different humans' biases on different reports' classifications. Even though our models cannot represent ground truth since such truth is not known, our models will enable consistent classification over time, which is necessary for trend analysis.

As an initial experiment, we looked at several multicategory classification techniques such as Naive-Bayes, Neural Networks, AdaBoost, and SVM [19]. The terms were ranked by Information Gain and the BOW matrix was reduced to the top 500, 1000, 2000, and 5000 terms. To evaluate the performance of the classifiers we used the area under the Receiver Operating Characteristic (ROC) curve. This is a curve that plots false positive rate (x-axis) against true positive rate (y-axis). The true positive rate is the fraction of documents inside a class that are correctly classified as being in the class. The false positive rate is the fraction of documents outside the class that are mistakenly classified as being in the class. Many classifiers return a number that indicates how likely an example is to be in a class. For example, a classifier may return a value from zero to one where one indicates that an example is predicted to be definitely in the class, whereas zero indicates that the example is definitely predicted to be out of the class. One may use a threshold of 0.5 on the returned value to decide whether an example is in the class. However, this threshold can be varied to achieve the desired trade-off between true positive and false positive rate. The values of true positive and false positive rates for these various threshold traces out the ROC curve. The best classifier achieves an area under ROC curve of 1, because there is at least one threshold for which it achieves a true positive rate of one and a false positive rate of zero.

As can be seen in Figure 5, standard SVM consistently matched or outperformed the other methods when looking at all categories. More recently, we observed some success using non-negative matrix factorization (NMF) and others have recently reported success with NMF for text mining as well [16]. In this paper we present our advancements of both SVM and NMF and demonstrate our ability to automatically categorize reports without the use of PLADS or other NLP methods for preprocessing.

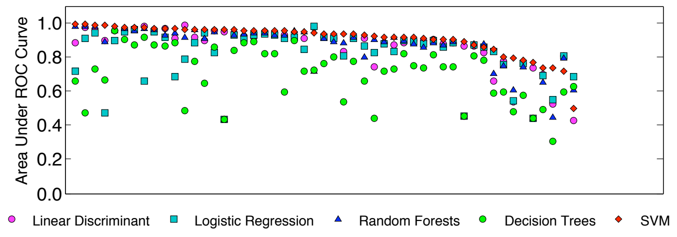


Fig. 5. Performance of different classification algorithms on 60 ASRS categories

Both the SVMs and NMF-based classifiers take as inputs documents in the form of a BOW matrix. The training set also contains labels for each document in the form of a matrix in which each row represents a document and each column represents a category. An entry contains a one if the corresponding document is thought to be an example of the corresponding category, and zero otherwise. Both SVMs and NMF-based classifiers were devised separately for each category, i.e., a one-vs-all methodology in which each classifier determines whether the document is in or out of its corresponding category. So in an operational setting where new documents are introduced, each category's classifier processes the document and predicts whether the document is in that category.

A. *Support Vector Machine (SVM)*

Support Vector Machines were developed by Vapnik in 1979 (see [7] for a tutorial). SVMs are a statistical learning method based on Structural Risk Minimization (SRM). In SRM, a set of classification functions or hypotheses that classify a set of data are chosen in such a way that minimizing the training error (what Vapnik refers to as the empirical risk) yields the minimum upper bound on the test error (what Vapnik refers to as the actual risk).

The simplest example of a Support Vector Machine is a linear hyperplane trained on data that is perfectly separable as shown on the left side of Figure 6. Given a set of input vectors, $\mathbf{x}_i \in \mathbf{R}^\mathbf{d}$, and labels, $y_i \in \{-1, 1\}$, SVM finds a hyperplane described by its normal vector, $\mathbf{w}$, and distance from the origin, b, that divides the data perfectly and is equidistant from at least one point in each class that is closest to the hyperplane (this distance is shown as $m/2$ in figure 6). This hyperplane is a decision boundary and the classification of an unknown input vector is determined by the sign of the vector operation

$$\mathbf{x}_i \cdot \mathbf{w} - b = d \tag{1}$$

If $d \geq 0$ $(d < 0)$ then the input is likely in the class $y = +1$ $(y = -1)$.

If the data are not perfectly separable this method can be adapted to compensate for instances that occur on the wrong side of the hyperplane. In that case slack variables, ξ_i (one for each training example), are introduced that measure the error of misclassified instances of the training data. SVMs find a hyperplane that best separates the data and minimizes the sum of the errors ξ_i by optimizing.

$$min \parallel \mathbf{w} \parallel + C \sum_i \xi_i \tag{2}$$

where C is a user defined weight on the slack variables. If C is large, the learning algorithm puts a large penalty on errors and will devise a more complicated model. If C is small, then the classifier is simpler but may have more errors on the training set. If the datasets are not balanced it is sometimes necessary to factor the errors of one class more than the other. Additional parameters, μ_y (one for each class), can be added to weigh one class error over the other.

$$min \parallel \mathbf{w} \parallel + \mu_{y_i} C \sum \xi \tag{3}$$

The decision hyperplane can only be linear which is not suitable for many types of data. To overcome this problem a function can be used to map the data into a higher or infinite dimensional space and run SVM on this new space.

$$\Phi : \mathbf{R} \mapsto \mathcal{H} \tag{4}$$

Because of the "kernel trick", $\mathcal{H}$ may have infinite dimension while still making the learning algorithm practical. In particular, one never works directly with Φ but rather with a kernel function K such that

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j). \tag{5}$$

There are many possible kernel functions available and new kernels can be built from the data itself. The kernels only need to meet Mercer's Conditions [8] to be used in SVM. For our classifier we chose a radial basis function kernel,

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2} \tag{6}$$

where γ is a parameter input by the user.

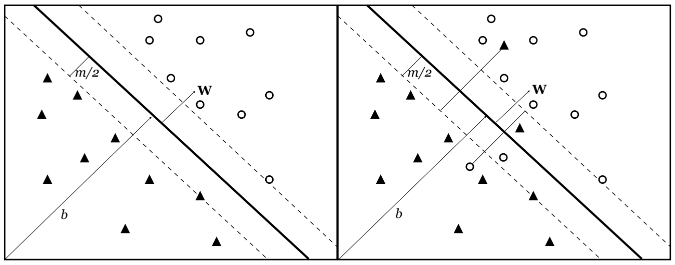


Fig. 6. Optimum Hyperplane for Separable and Non-Separable Case

1) Hyperparameters: For the set of aviation safety documents, we have found that the values of the hyperparameters, $C, \mu,\ and \gamma$, greatly affect the performance of the classifier. The number of dimensions (i.e., terms), unknown relationships between categories, and the different ratios of documents in each class make it difficult to build a general set of parameters. So, they are either determined from basic tests on the input data or found by a simple grid search. In practice, C and γ are determined by starting with a coarse mesh and iterating at increasingly finer scales. This can take hundreds of evaluations for just two hyperparameters. When adding the μ parameter to compensate for the unbalanced data, it can take thousands of iterations to complete even a coarse grid. Instead of using a brute force search over a grid, we used a Simulated Annealing search to determine the optimum hyper-parameters. We refer to the combined SVM and Simulated Annealing Algorithm as Mariana in honor of the Mariana Trench which is the deepest location on the surface of the Earth's crust and as such "has achieved the minimum value."

2) Simulated Annealing: Simulated Annealing is a stochastic search optimization similar to Markov Chain Monte Carlo [9]. The basic algorithm evaluates the output of the classifier for a given set of parameters and then randomly adjusts the values of the hyperparameters for reevaluation. If the new set of hyperparameters improves the results, the new set is kept. If it does not, then the results are kept or rejected based on the probability of how different the current result is from the best result thus far. Modifying the hyperparameters occasionally even when they are moving away from the current maximum reduces the probability of settling on a local maximum, increasing the chance of achieving the global maximum. The data set is broken into three parts, 50% training, 25% validation, and 25% test. The natural distribution of the categories is maintained in each part. Since we are treating the multicategory problem as a set of binary classification problems, there are significantly fewer documents in each class than not in the class. To ensure that there are enough positive examples, the training file for each binary classifier is built by randomly selecting document vectors from the training set while maintaining a minimum of 10% of documents in the class. The validation and test set do not change. Once a model is built from the training set, it is used to predict the categories of the validation set. We use the area under the ROC curve to evaluate the predictions. This process is repeated until a maximum is found or for a predefined number of iterations. Once the optimum hyperparameters are found for each class the model is then used for a final prediction of the test data.

3) Confidences: The output from the prediction is the distance from the hyperplane and can be any real number. This can be misleading when comparing the outputs from different class models for the same document because the scales of the different class models need not be the same. Even if the outputs are comparable in range, the distance is not a true measure of confidence in the prediction. The categorization is determined by the sign of the output from the SVM and it traditionally discards the distance. We combine the distance and the rate at which each category occurs in the data set to generate a confidence as follows:

$$Conf = \frac{1}{1 - exp\left(-\alpha d + \beta\right)}, \tag{7}$$

where α and β contain category information. The final output is a positive number between 0 and 1 and scaled; so any value above 0.5 indicates that the document is predicted to be in the class.

V. NMF for ASRS Text Categorization

Non-negative Matrix Factorization (NMF) is a variation on the host of mathematically motivated techniques for factoring large vector-valued data arrays into basis and distribution matrices. Suppose we have d documents and t terms[1]. The general approach is to seek a relatively small set of k basis vectors represented by the t x k matrix W, and a corresponding set of distribution weight vectors represented by the k x d matrix H, such that the transposed bag-of-words matrix X (for NMF, we assume X is k x n rather than the usual convention of being n x k) is factored according to $X \approx W * H$ by minimizing some measure of the difference, $X - W * H$. The hope is that the basis vectors will correspond to some fundamental properties of the data set, with the distributions mapping those properties to the data.

Regarding the form ($X \approx W * H$), the convention in NMF-based text analysis is that the jth column of X represents the term weights of document j, each column of W is a basis vector over the term set, and each column of H is is a set of weightings over the basis vectors. Thus, if we use $H_{\bullet j}$ to denote the jth column of H and $H_{i\bullet}$ to denote the ith row of H, then for the jth document, $W * H_{\bullet j} \approx X_{\bullet j}$. If each basis vector $W_{\bullet b}$ is individually L1 normalized to a single term weight, and the rows $H_{b\bullet}$ are inversely scaled to maintain the product, then H_{bj} is the approximate number of terms, from basis vector $W_{\bullet b}$, found in the jth document. If the columns $H_{\bullet j}$ are then L1 normalized, they give the relative basis vector weights, independent of document size.

In NMF applications, the data values are non-negative, typically counts or scalar measurements, and the factorization is constrained to keep both W and H non-negative. This has a strong appeal lacking in factorization methods that allow either basis or distribution values to be negative. With non-negativity, the basis vectors may be thought of as components, and the distributions as recipes for adding components to match the data. Both aspects seem more natural than the alternatives, particularly so for intrinsically non-negative data.

While non-negativity is an appealing property for factorization methods, constraining conventional difference minimization algorithms to maintain non-negativity has mostly been difficult. This changed circa 2002, with spreading recognition of the potential of Lee and Seung's multiplicative update approach [12], [13]. For the squared Frobenius norm, the standard sum of squared matrix values, Lee and Seung's original paper gives the minimizing reestimation relations as:

$$W_{ab} = W_{ab}\frac{(X * H^T)_{ab}}{(W * H * H^T)_{ab}} \tag{8}$$

$$H_{bi} = H_{bi}\frac{(W^T * X)_{bi}}{(W^T * W * H)_{bi}} \tag{9}$$

where a and i index over the attributes and instances of X, respectively, and b indexes over the basis vectors. This is actually a reformulation of the standard gradient driven norm minimizing search, augmented with a conceptually simple step size computation that maintains the non-negativity constraint.

Starting with non-negative W and H, and applied alternately, these reestimation equations are proven to monotonically lower the norm toward a local stationary point, while maintaining the non-negative properties of W and H. Lee and Seung also provided an alternate multiplicative minimization for the Kullback-Liebler divergence of probability matrices, and [14], [15] have since developed versions for other matrix norms.

A. NMF Practicalities

In factoring the prepared data, there is the fundamental choice of what difference matrix norm shall be minimized with consequences that are not yet well understood. A basis size must be chosen, or a range of sizes searched over and evaluated. Algorithmic details, particularly factor initializations, may have significant effects. Since the multiplicative NMF algorithms are gradient driven, they approach their stationary points at exponentially decreasing rates. This requires stopping criteria that balance the opposing requirements of computational efficiency and numerical accuracy.

Once the data has been factored, understanding the basis and distribution can be challenging. This is a major problem for undirected data mining. Understanding is largely a matter of relating results to external criteria, and so requires additional information, e.g., subject expertise. In supervised learning, we have the advantage of a standard against which we can rate our results.

Our current system uses the standard English stopword list[2] to eliminate common English terms that would appear in a large percentage of the documents and therefore would be useless for classification. Terms unique to any single document are clearly useless, although surprisingly common in the ASRS narratives. A variety of term count weighting schemes have been devised, intended to emphasize some aspect of the

[1]in general, NMF can be used in the case where there are d examples that have t features each, but we present NMF as we are using it for text classification.

[2]SMART's English stoplist at ftp://ftp.ce.cornell.edu/pub/smart/english.stop

statistics. We have found that term weightings which improve classification performance may also greatly increase the error norm that our NMF algorithm minimizes, and vice versa, so end-to-end testing is essential.

B. NMF Application

The work described here was inspired by the promising results shown in [16] which came in second place in the 2007 SIAM Text Mining Competition, which used the ASRS data that we use in this paper. Our NMF work described in this paper is a follow on of that entry, aimed first at reproducing their results, and then exploring the effects of alternatives in weighting, factorization and classifier construction. So we begin by describing their approach and results.

For term extraction, they applied their General Text Parser, accepting as terms any strings of 2 to 200 characters, which appear in 2 or more reports. These were filtered against the stop word list, giving 15,722 terms from 21,519 documents. For each term t, they compute an entropy with respect to the document set as $g(t) = 1 + \sum_d (p(t,d) * log_2(p(t,d)))/log_2(D)$, where $p(t,d)$ is the observed proportion of term t in document d, over the D documents. Thus $g(t)$ ranges from 1, for a term appearing in a single document, to 0, for one that is uniformly distributed over all documents. Their final weight for occurrence count $X(t,d)$ is $g(t) * log_2(1 + X(t,d))$, thus de-weighting large occurrence counts. No within-document normalization was applied.

Their factorization used the original Lee and Seung algorithm for minimizing the summed squared difference, $W * H - X$, started from random W and H matrices, with 5 convergence cycles. Their basis size was set at 40, almost twice the category count of 22. Their predictive system was tuned against the training data by splitting 70-30 for training and testing. In this, their entropic weight, $g(t)$, forces a choice regarding what data will contribute. Their choice was to group both training and test documents (not labels), compute $g(t)$ with respect to the whole, factor the whole, and then separate the distribution, H, into training and test sets for prediction. The same grouped approach was used for their contest entry, with only training categories available.

Their predictor construction is fundamentally

$$C_p = f(H_p^T * (H_t * C_t)) \tag{10}$$

where inputs H_t and H_p are the NMF training and predictive distribution matrices, and C_t and C_p are 0/1 valued training and predictive document category indicator matrices. Function f combines a level cutoff and binary conversion from assessments to logical predictions. In practice, several other filters are applied to the inputs and intermediate results.

C. NMF Investigations

We used our own term extraction code, with the same stop word list, getting 15431 two or more character terms appearing in two or more documents of the training set—slightly less than what was found in [16]. While placing no limit on term length, we found none that exceeded the 200 character limit used in [16]. Our NMF code has evolved from the public distribution of Dr. Patrick Hoyer[3], but was run in the same basic mode used in [16], with identical core computations. We typically ran the convergence to 50 cycles, well down on the exponential approach to convergence, or else to a 1e-4 relative convergence rate, which normally terminated between 70 and 100 cycles.

Our classifier construction code is a MATLAB memory limit adaptive matrix coded reimplementation of that submitted to the SIAM competition by Allen, et. al.,[16]. Predictive quality is evaluated per the SIAM competition specification, basically a sum of the traditional receiver operator characteristic (ROC) curve area and a category confidence measure. Using these, we have duplicated the results in [16], improved on that result by 25%, and investigated a range of variations on their approach to NMF based classification.

We initially investigated two simple alternatives to the term weighting used in [16]. Using the raw counts, as unweighted term counts, reduced the NMF error matrix norm about 30-fold, relative to the log-entropic weighted terms cited in [16], but severely degraded prediction quality. Using raw counts with stop words retained gives about a 100-fold reduction in NMF norm, and very severe reduction in prediction quality. A later survey, using the ASAP data to predict contributing factors, looked into all 24 combinations of the 4 local and 6 global weightings discussed in [6]. None of these combinations showed any significant improvement over the log-entropy weighing.

We investigated the effect of retaining only terms that appear in a minimum number of documents. We have investigated minimum document counts of 2, 5, 10, 20, and 40. At a minimum of 40 documents per term, we eliminate about 90% of the original terms, while retaining about 90% of the original term occurrence counts. Thus there are clear computational advantages to term filtering. We find a clear response to term filtering in the NMF approximation error norm, with a minimum Frobenius norm consistently observed at 10 documents per term. This optimality of the factorization error is not reflected in the final predictive performance. Seemingly random variations between otherwise identical runs, attributed to the random NMF factor initialization, generally exceed the variation between the minimum document count groups. Increasing minimum document count over the range from 2 to 40 increases the predictive quality by only a few percent, but greatly reduces computational costs.

We made a limited investigation into the effect of varying the NMF basis size, k, choosing k of 33, 44, 55 and 99 as reasonable for a category count of 22. At these levels, increasing basis size gives a small but consistent decrease in the error matrix norm and a corresponding increase of a few percent in the prediction quality, with proportional increase in computational costs.

The Lee and Seung algorithm's error matrix norm, plotted against convergence cycles, shows the exponential approach to a fixed point that is characteristic of many difference-driven algorithms. We have investigated the effect of this convergence on prediction quality, recording the factorization at every

[3]http://www.cs.helsinki.fi/patrik.hoyer/. A related paper is [18].

5 cycles. The initial convergence actually gives reasonable predictions, with quite good predictions at 5 cycles, the quality increasing an additional 30% between 5 and 30 cycles, and only slightly thereafter. Increasing convergence cycles over the 5 used in [16] was the chief source of our improved results, but entails significant computational costs for rapidly decreasing improvement.

The predictor construction in [16] involves several document and category-wise relative magnitude filters, and conversion of the filtered H_t to binary values. Investigating the parameter space, we find predictive quality is not strongly affected, so long as the final document-wise relative thresholding of C_p[4], for acceptance of the larger category values as assignments, is not too low. Eliminating the binary conversion of H_t does help slightly.

The core predictive computation done in [16] is essentially

$$C_p = H_p^T * H_t * C_t.$$

Symmetry suggests that $H_p * C_p = H_t * C_t$ should give better predictions, preferably as $C_p = H_p^{-1} * H_t * C_t$. Unfortunately, H_p is very far from being invertible. We investigated ways to solve the symmetric form for non-negative C_p. Our best approach uses a few half cycles of Lee and Seung's multiplicative update, solving $H_p * C_p = H_t * C_t$ with H_p and $H_t * C_t$ held fixed throughout. The results, with respect to variation of other factors, are generally consistent with those obtained in [16]. With random initialization of C_p, the predictive quality is depressed about 10% relative to Allen et al. [16] Initializing C_p with Allen et al.'s result, and dropping the filters on H_t and H_p, gives about 25% improvement on their reported results.

In [16], the entropic term weight $g(t)$ sums $p(t,d)log_2(p(t,d))$ over all documents d. This works well when both training and test documents are at hand. For a production system, it will be advantageous to train the system on initial data, and then apply it to new instances as they become available. It is then necessary to retain $g(t)$, possibly update it, and apply it to new instances. We have not investigated this evolution of $g(t)$, using the fixed value from the training data for our production mode experiments.

Obtaining a classifier in production mode is more difficult than it was for Allen et al.[16], where they factored the joint training plus test data set and then separate H into H_t and H_p to classify the test set using $C_p = H_p^T * H_t * C_t$. To duplicate the situation in production mode where one only has a training set to work with and then has to classify test sets without seeing the narratives first, we factored the training BOW X_t into $W_t * H_t$, and then obtained H_p by solving the equation $W_t * H_p = X_p$. We then classified test data using $C_p = H_p^T * H_t * C_t$. The predictive quality is only slightly lower than that obtained with joint factoring. This is as expected, on account of the smaller data set used to generate W_t, which only has 3/4 the document set and 90% of the terms available in the joint calculation. We also performed some limited experiments on ASAP in which we performed the factorization with both the labeled training documents and the unlabeled documents, which is an example of semi-supervised learning. The performance did not improve significantly over the case where we did not use the unclassified documents.

[4]For each document, a threshold is chosen such that for each category, if the corresponding value of C_p is greater than the threshold, then the document is considered to be a member of that category, and otherwise it is not.

VI. Results

We first tested Mariana with raw and PLADS-processed text to assess how useful PLADS and the simulated annealing within Mariana would be. Mariana's process of choosing better hyperparameters improves the area under the ROC curve by 10% or more in some of the categories on raw text. It also gives equal improvement on text that has been processed by the PLADS system. As shown in Figure 7, there is little difference in performance when this process is used on raw text and PLADS processed text. Because of the overhead NLP techniques tend to require. For this reason, in our subsequent tests, we use only raw text.

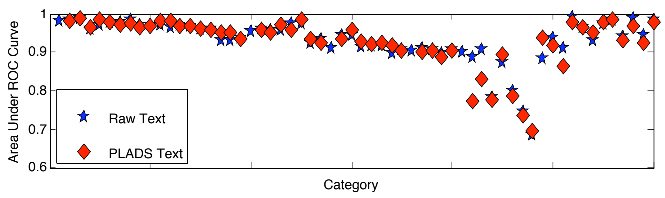


Fig. 7. Comparison of Raw Text and PLADS Processed Text using Optimum Hyperparameters

Once convinced that the algorithms were comparing well against other techniques, we evaluated the performance of the full tool by auto-classifying 100 randomly selected reports and having the results reviewed by a problem report expert. The reviewed results were encouraging. Our stated goal was for the reviewer to approve of at least one of the auto-classification selections 75% of the time. The reviewer agreed with the top classification 73% of the time. The reviewer agreed with one or both of the top two classifications 86% of the time, and with the top three classifications 90% of the time. A separate review of the 100 reports was done by another subject expert. Then our reviewer reviewed their classifications, and agreed with their top (and only) classification 89% of the time. These results not only indicate that Mariana is well within expected classification levels, but that reviewers disagree amongst themselves. Since there is no expectation that an automated learning system could outperform the expert who classified the training documents that it learns from, we feel that the Mariana results are quite positive.

We include the results of applying Linear Discriminant Analysis [10], which is a linear classification method, in order to assess whether Mariana and NMF are generating a model more sophisticated than a linear model. Linear Discriminant Analysis assumes that each class is modeled as a Gaussian in the space of its features and finds a linear classifier that separate the classes.

Figure 8 shows the areas under the ROC curves for Mariana, NMF, and LDA applied to the ASRS event types. Overall,

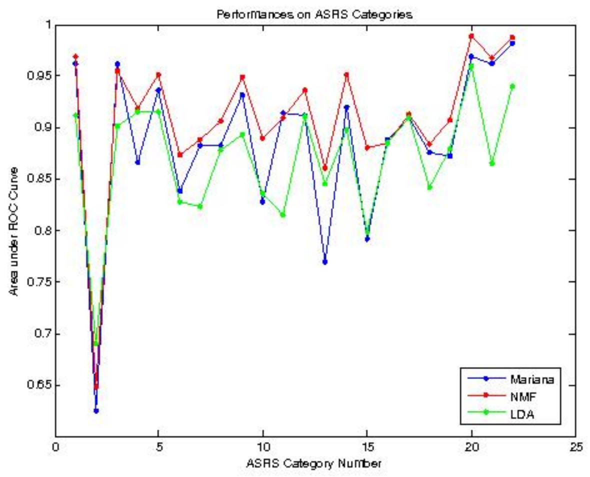


Fig. 8. Performances of Mariana and NMF on ASRS categories.

Mariana and NMF outperform LDA most of the time—Mariana outperforms LDA on 16 of the 22 categories and NMF outperforms LDA on 21 of the 22 categories—however, not by a great amount.

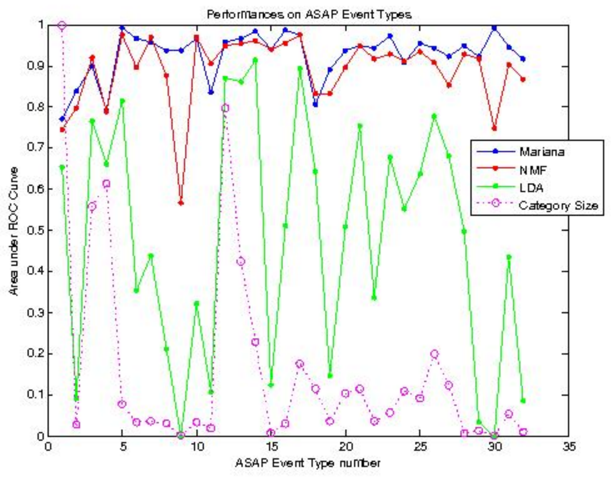


Fig. 9. Performances of Mariana and NMF on ASAP Event Types.

Figure 9 shows the areas under the ROC curves for Mariana, NMF, and LDA applied to the ASAP event types, together with the fraction of documents in each category, normalized so that the category with the largest number of documents (category 1) has a value of 1. We see that LDA tends to perform particularly poorly when the number of documents in the category is low. NMF and especially Mariana are much more stable in their performances and clearly outperform LDA. The stability of performance of NMF and Mariana and their good performances on low-population categories is very important for classifying ASRS and ASAP reports since we expect many categories to have few documents but accurately determining when a document is in one of these categories is very important.

Figure 10 shows the areas under the ROC curves for Mariana, NMF, and LDA for the ASAP contributing factors, as well as the fraction of documents in each category normalized so that the category with the largest number of documents has value 1. LDA's performance is more stable here than for event types because the contributing factors all seem to have a reasonable number of documents, whereas several event types have too few documents for LDA to give a useful result. Once again, Mariana and NMF show superior performance relative to LDA, especially on contributing factors that have relatively few documents.

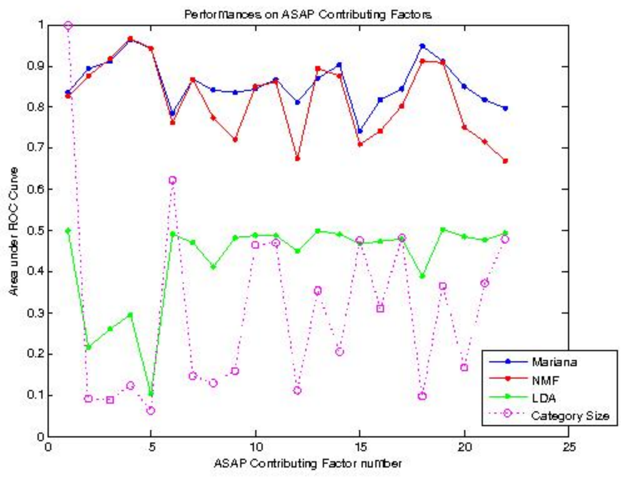


Fig. 10. Performances of Mariana and NMF on ASAP Contributing Factors.

In section 5, we pointed out that one of the appealing properties of NMF is that it produces nonnegative components of the data being factored. These components have sometimes shown themselves to correspond to realistic parts of the objects being considered, such as parts of a human face [15]. We have begun investigating the interpretability of the basis vectors produced by NMF on our text classification problems. In Table IX, we give examples of two basis vectors from three runs of NMF (with different training and test set mixes drawn from the original dataset) on the ASRS repository. One basis vector was drawn from each of the three runs such that the resulting three basis vectors were the closest such triple in terms of $L1$-norm. The top 20 words within the bases in terms of weight are shown as the three left columns of table IX. The three right columns are the words corresponding to the second closest such triple. One can see from these examples that the three runs came up with bases that are relatively close to one another. The two triples are also quite different from each other. In examining six such triples of basis vectors, we found that the six bases were quite different from one another. However, the 20 words within a basis vector are clearly quite related. The first basis set clearly seems to be describing some fuel tank related problem and the second basis set clearly describes an issue related to a repair. We are unable to show corresponding results on the ASAP data because they are proprietary; however, we did observe basis sets with similar properties in that the bases were significantly different from each other but the words within each basis were clearly related. We mentioned in section 2 that the BOW representation maintains word collocation information. It appears that NMF is finding the significant collocations that are indicative of topics and keeping them together.

VII. Mariana - The Tool

We have incorporated Mariana into a tool that allows analysts to view text reports and Mariana's classifications of them, and also provide feedback on the classifications. We have built an interactive web tool that incorporates the Mariana models and classifier with an online database of ASRS reports. The flowchart in Figure 11 describes the process.

The subject expert analysts can log onto the website and select a set of reports to be auto-classified. Once a set of reports are selected, Mariana will predict the categories the documents belong to and display only the predicted categories that have a 50% confidence or more. The user can then evaluate the report and confirm or correct the classifications. Figure 12 is a screenshot of the tool displaying a sample report, and the results of the Mariana classification algorithm. And, in fact, as can be seen in the figure, the classifications are ranked by confidence in the prediction. The Mariana auto-classification online system is going through a trial at an air carrier's site.

The reports can be displayed to an unlimited number of reviewers and each of the expert's classifications are kept, thus providing an ability to track both the performance of the auto-classification models and algorithm, and the inter-operator agreement. New models can be easily built and incorporated into the tool, which would be useful when new data is available or the data changes enough that the current model's performance drops.

VIII. Future Work

The tool that we presented above is designed to allow expert user feedback on the classifications done by the model. To this end, we will investigate online updating of our models as new reports arrive. We also plan to incorporate NMF and other models into the tool depicted in figure 12 to form an ensemble classifier which can outperform any of its constituent models individually if the appropriate combining scheme is used. Within the SVM and NMF-based classifiers themselves, we plan to investigate how to enable them to directly optimize false positive rate and false negative rate at the desired tradeoff (e.g., optimize false positive rate subject to a maximum allowable false negative rate of 0.01), rather than having to optimize accuracy (for SVM) or a matrix norm (for NMF) and hope that the result yields a good area under the ROC curve.

We plan to use the tool depicted in figure 12 as well as any classifiers that we add to it to perform trend analysis of the various anomaly categories. Within ASAP we plan to further investigate possible connections between the contributing factors and event types as we are able to leverage more reports and perhaps other relevant data.

IX. Conclusion

In this paper, we presented the results of classifying ASRS and ASAP text reports using an SVM-based classifier and an NMF-based classifier. We showed that the accuracies of these algorithms are quite high and are relatively stable over a set of categories with highly-varying numbers of reports. We also demonstrated the potential for NMF bases to be interpretable and allow identification of significant topics within a document repository. We have also developed a tool that presents classification results in an intuitive manner that allows experts to view these classifications and provide feedback. The tool, together with the classifiers we have developed so far and new classifiers that we and others will develop, will enable trend analysis of the various anomaly categories and; therefore, a greater ability to characterize the health of the national airspace.

References

[1] NASA, "Aviation Safety Reporting System (ASRS) Program Overview," http://asrs.arc.nasa.gov/overview.htm, 2007.

[2] A.N. Srivastava and B. Zane-Ulman, "Discovering Recurring Anomalies in Text Reports Regarding Complex Space Systems," Proceedings of the 2005 IEEE Aerospace Conference, 2005.

[3] A.N. Srivastava, et al., "Enabling the Discovery of Recurring Anomalies in Aerospace Proplem Reports using High-Dimensional Clustering Techniques," 2006 IEEE Aerospace Conference, Big Sky, MT, March 2006.

[4] MATLAB's single linkage and agglomerative clustering.

[5] A. Banerjee, et al., "Generative Model-based Clustering of Directional Data," SIGKDD '03, Washington, DC, USA, August 2003.

[6] M.W.Berry and MBrowne, Understanding Search Engines - Mathematical Modeling and Text Retrieval, SIAM, 1999.

[7] C.J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery, 2(2):955-974, 1998.

[8] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines and other Kernel-Based Learning Methods, Cambridge University Press, 2000.

[9] R. Duda, P. Hart, and D. Stork, Pattern Classification, 2nd ed., John Wiley & Sons, 2001.

[10] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Springer, 2001.

[11] D.D. Lee and H.S.Seung, "Learning the parts of objects by non-negative matrix factorization.", Nature, 401(6755):788-791, 1999.

[12] D.D. Lee and H.S.Seung, "Algorithms for Non-negative Matrix Factorization", in Advances in Neural Information Processing 13 (Proc. NIPS 2000), MIT Press, 2001.

[13] L.K.Saul and D.D. Lee, "Multiplicative Updates for Classification by Mixture Models", Proceedings NIPS 2001.

[14] I.S.Dhillon and S.Sra, "Generalized Nonnegative Matrix Approximations with Bregman Divergences", Proceedings NIPS 2005.

[15] Y.Wang, Y.Jia, C.Hu and M.Turk, "Fisher Non-Negative Matrix Factorization for Learning Local Features", in Asian Conference on Computer Vision, 2004.

[16] E.G. Allen, M.R. Horvath, C.V. Kopek, B.T. Lamb, T.S. Whaples and M.W. Berry, "Anomaly Detection Using Non-negative Matrix Factorization", Survey of Text Mining II, pp. 203-217, Springer London, 2007.

[17] Seventh SIAM International Conference on Data Mining (SDM 2007), Text Mining Workshop's text mining competition website at http://www.cs.utk.edu/tmw07/.

[18] P. Hoyer, "Non-negative matrix factorization with sparseness constraints", in Journal of Machine Learning Research 5:1457-1469, 2004.

[19] A.N. Srivastava, et. al., "Enabling the Discovery of Recurring Anomalies in Aerospace Problem Reports using High-Dimensional Clustering Techniques," Proceedings of the IEEE Aerospace Conference, Big Sky, Montana, March 4-11, 2006.

[20] A. Hotho and A. Nurnberger and G. Paab, "A Brief Survey of Text Mining," Zeitschrift fuer Computerlinguistik und Sprachtechnologie (GLDV-Journal for Computational Linguistics and Language Technologie). 20(2):19-62, 2005.

TABLE IX
EXAMPLES OF ASRS BASIS VECTORS

FUEL	FUEL	FUEL	INSTALL	INSTALL	INSTALL
TANK	TANK	TANK	INSPECT	INSPECT	REMOVE
POUND	POUND	POUND	REMOVE	REMOVE	REPLACE
GALLON	GALLON	GALLON	REPLACE	MECHANIC	ENGINEER
GAUGE	GAUGE	GAUGE	MECHANIC	REPLACE	MANUAL
PUMP	PUMP	PUMP	FOUND	PART	INSPECT
FUELTANK	BURN	BURN	WORK	MANUAL	WORK
BURN	FUELTANK	FUELTANK	MANUAL	WORK	SHIFT
FUELER	FUELER	FUELER	REPAIR	REPAIR	FOUND
FUELQUANTITY	FUELQUANTITY	FUELQUANTITY	PART	FOUND	ASSEMBLE
CENTER	CENTER	CENTER	ENGINEER	SIGN	TECHNICIAN
MAINTANK	DISPATCH	FUELGAUGE	TEST	ENGINEER	REPORT
FUELGAUGE	FUELGAUGE	MAINTANK	CHECK	NUMBER	PANEL
IMBAL	MAINTANK	IMBAL	SHIFT	SHIFT	REPAIR
REFUEL	IMBAL	REFUEL	SIGN	MAINTAIN	JOB
CROSSFEED	REFUEL	PLAN	ASSEMBLE	TEST	XYZ
QUANTITY	QUANTITY	CALCULATE	MAINTAIN	ASSEMBLE	BOLT
BALANCE	PLAN	CROSSFEED	SERVE	AIRCRAFT	CARD
CALCULATE	CROSSFEED	BALANCE	CARD	XYZ	LEAK
EMPTY	CALCULATE	EMPTY	TECHNICIAN	TECHNICIAN	JOBCARD

[21] S. Hettich and S.D. Bay, ”The UCI KDD Archive [http://kdd.ics.uci.edu],” Irvine, CA: University of California, Department of Information and Computer Science, 1999.

[22] S. Wolfe, ”Wordplay: An Examination of Semantic Approaches to Classify Safety Reports,” AIAA Infotech@Aerospace, AIAA Paper 2007-2821, Rohnert Park, CA, May 2007.

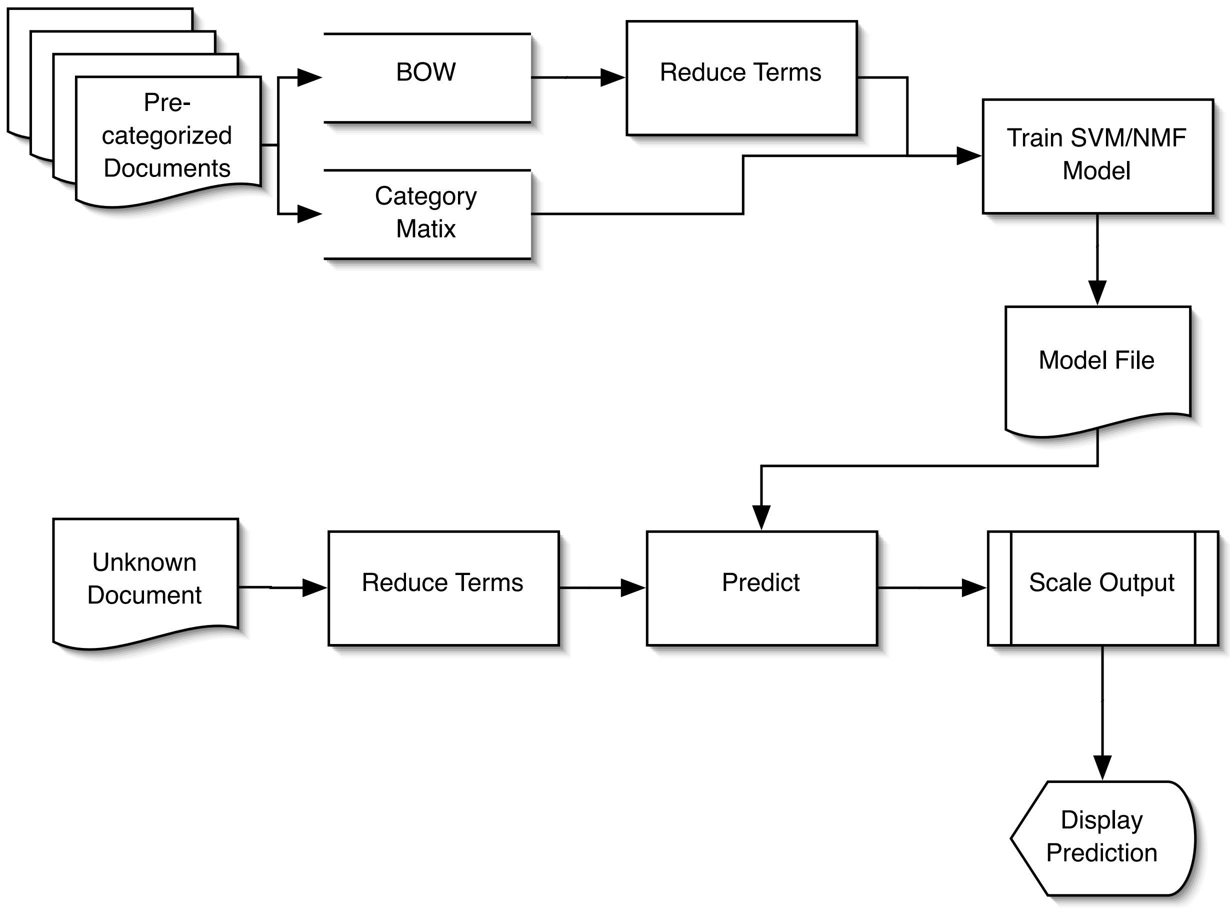
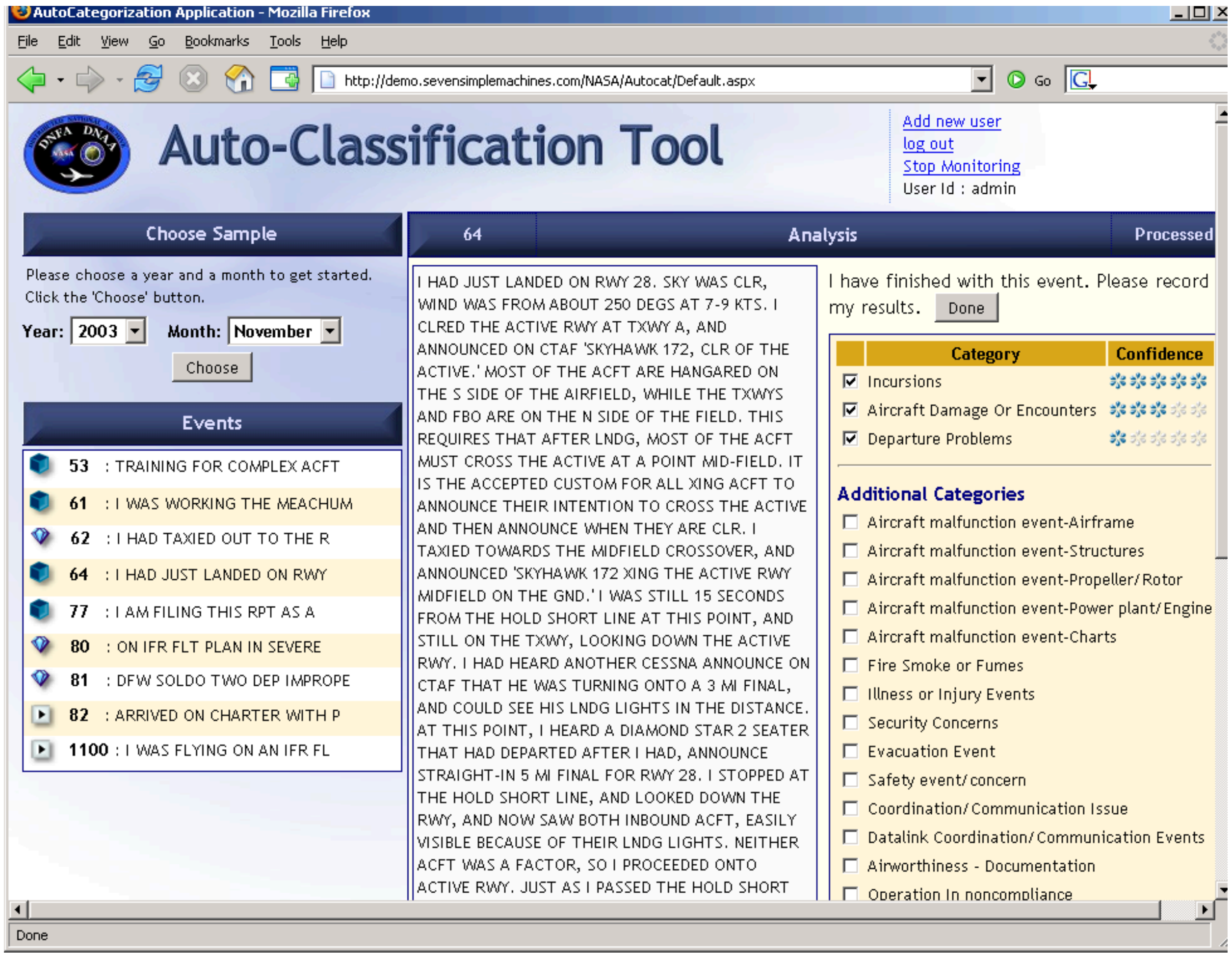


Fig. 11. ASAP Classification Flow in Mariana


AutoCategorization Application - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://demo.sevensimplemachines.com/NASA/Autocat/Default.aspx
Go

Auto-Classification Tool

Add new user
log out
Stop Monitoring
User Id : admin

Choose Sample

Please choose a year and a month to get started. Click the 'Choose' button.

Year: 2003 Month: November

Choose

Events

53	: TRAINING FOR COMPLEX ACFT
61	: I WAS WORKING THE MEACHUM
62	: I HAD TAXIED OUT TO THE R
64	: I HAD JUST LANDED ON RWY
77	: I AM FILING THIS RPT AS A
80	: ON IFR FLT PLAN IN SEVERE
81	: DFW SOLDO TWO DEP IMPROPE
82	: ARRIVED ON CHARTER WITH P
1100	: I WAS FLYING ON AN IFR FL

64 **Analysis** Processed

I HAD JUST LANDED ON RWY 28. SKY WAS CLR, WIND WAS FROM ABOUT 250 DEGS AT 7-9 KTS. I CLRED THE ACTIVE RWY AT TXWY A, AND ANNOUNCED ON CTAF 'SKYHAWK 172, CLR OF THE ACTIVE.' MOST OF THE ACFT ARE HANGARED ON THE S SIDE OF THE AIRFIELD, WHILE THE TXWYS AND FBO ARE ON THE N SIDE OF THE FIELD. THIS REQUIRES THAT AFTER LNDG, MOST OF THE ACFT MUST CROSS THE ACTIVE AT A POINT MID-FIELD. IT IS THE ACCEPTED CUSTOM FOR ALL XING ACFT TO ANNOUNCE THEIR INTENTION TO CROSS THE ACTIVE AND THEN ANNOUNCE WHEN THEY ARE CLR. I TAXIED TOWARDS THE MIDFIELD CROSSOVER, AND ANNOUNCED 'SKYHAWK 172 XING THE ACTIVE RWY MIDFIELD ON THE GND.' I WAS STILL 15 SECONDS FROM THE HOLD SHORT LINE AT THIS POINT, AND STILL ON THE TXWY, LOOKING DOWN THE ACTIVE RWY. I HAD HEARD ANOTHER CESSNA ANNOUNCE ON CTAF THAT HE WAS TURNING ONTO A 3 MI FINAL, AND COULD SEE HIS LNDG LIGHTS IN THE DISTANCE. AT THIS POINT, I HEARD A DIAMOND STAR 2 SEATER THAT HAD DEPARTED AFTER I HAD, ANNOUNCE STRAIGHT-IN 5 MI FINAL FOR RWY 28. I STOPPED AT THE HOLD SHORT LINE, AND LOOKED DOWN THE RWY, AND NOW SAW BOTH INBOUND ACFT, EASILY VISIBLE BECAUSE OF THEIR LNDG LIGHTS. NEITHER ACFT WAS A FACTOR, SO I PROCEEDED ONTO ACTIVE RWY. JUST AS I PASSED THE HOLD SHORT

I have finished with this event. Please record my results. Done

Category	Confidence
☑ Incursions	★★★★★
☑ Aircraft Damage Or Encounters	★★★☆☆
☑ Departure Problems	★☆☆☆☆

Additional Categories

- ☐ Aircraft malfunction event-Airframe
- ☐ Aircraft malfunction event-Structures
- ☐ Aircraft malfunction event-Propeller/Rotor
- ☐ Aircraft malfunction event-Power plant/Engine
- ☐ Aircraft malfunction event-Charts
- ☐ Fire Smoke or Fumes
- ☐ Illness or Injury Events
- ☐ Security Concerns
- ☐ Evacuation Event
- ☐ Safety event/concern
- ☐ Coordination/Communication Issue
- ☐ Datalink Coordination/Communication Events
- ☐ Airworthiness - Documentation
- ☐ Operation In noncompliance

Done

Fig. 12. Screenshot of Auto-Classification Tool