Automatic document filing according to user categories

Dong CAO* Judith GELERNTER Jaime CARBONELL

Carnegie Mellon University
Pittsburgh, PA U.S.A.

* Also, Beijing University of Posts and Telecommunications

Abstract
Suppose you have just acquired a number of articles into a personal digital library. The classification procedures described here would allow those articles to be automatically sorted in your pre-existing desktop folders. We use the Chi-square statistic to find keywords that we use to classify the articles, and the Support Vector Classifier to organize the articles in folders. Moreover we have adapted the process so that minimal feedback from users should improve classification results, and folders do not contain overly many articles for information management convenience. We achieved an average of 96% accuracy over 10 trials with the Reuters 21578 news data, and 89% accuracy over 10 trials with biomedical literature downloaded from the PubMed digital library. We conclude that in a personal desktop environment, whether general or highly specific, our classification methods for automatic filing into the user's own familiar folders would achieve high accuracy.

Keywords: classification, personal digital document management, personal information management, document organization, digital library, SVM, Chi square, text categorization

I. Introduction

A study on information preferences found that users like to search for a document by location, that is, the location on their own computer, so that they do not need to remember the file name (Barreau and Nardi, 1995). Our procedures would allow users to keep their familiar categories and acquire documents en masse, to be filed automatically. Automatically filing stands to be become even more useful in the near future as consumers might be acquiring music, articles, photos, and video on a single device (ICT, 2010).

The challenge of automatic filing was presented to us by the director of the personal information management software, Mendeley. [1] He told us that his users requested automatic filing such as is found in the MacIntosh Smart Folder or the iTunes Smart Playlist. We know that users tend to assign filing categories that mix genre, category, task and time (Henderson, 2005), and that retrieval by date or media is straightforward. As a research project, we focus on automatic filing by subject. We use Mendeley as an environment for a user scenario.

Mendeley software stores articles and provides its own categories for organization. Over 120,000 downloads of the software were counted as of January 2010, and there are almost 100,000 registered users. Our procedure will allow users to file unseen documents automatically into user-defined categories. The same procedure could be used

[1] Dr. Jason Hoyt, Director of the Academic Reference Management software Mendeley, at http://www.mendeley.com/

to organize an entire collection of articles into Mendeley category for storing on the back end on the company's server.[2]

Consider this use case to demonstrate how our auto-file system would work. A Mendeley customer stores his articles in category folders for Artificial Intelligence, Biochemistry, Bioinformatics, Experimental psychology, Genetics, Molecular biology and Neurobiology. A screen shot of the software desktop, left-hand column, shows these labelled folders (Figure 1). The customer, a scientist, has just downloaded a set of articles sent by one of his colleagues. These articles are as yet unsorted, and most are unread, as indicated by the un-bold dot preceding the article title (see right pane of screen in Figure 1). The addition of our classification procedures would tuck the new articles into the existing folder categories. Each article is assigned one category only.

The articles in the user's folders are used as training data. The more articles the user has in a folder, the more likely that new articles will be classified correctly. Our procedure uses data mining to find keywords, statistics to prioritize key words, and machine learning to acquire the algorithm that will predict in which categories unseen articles should be filed.

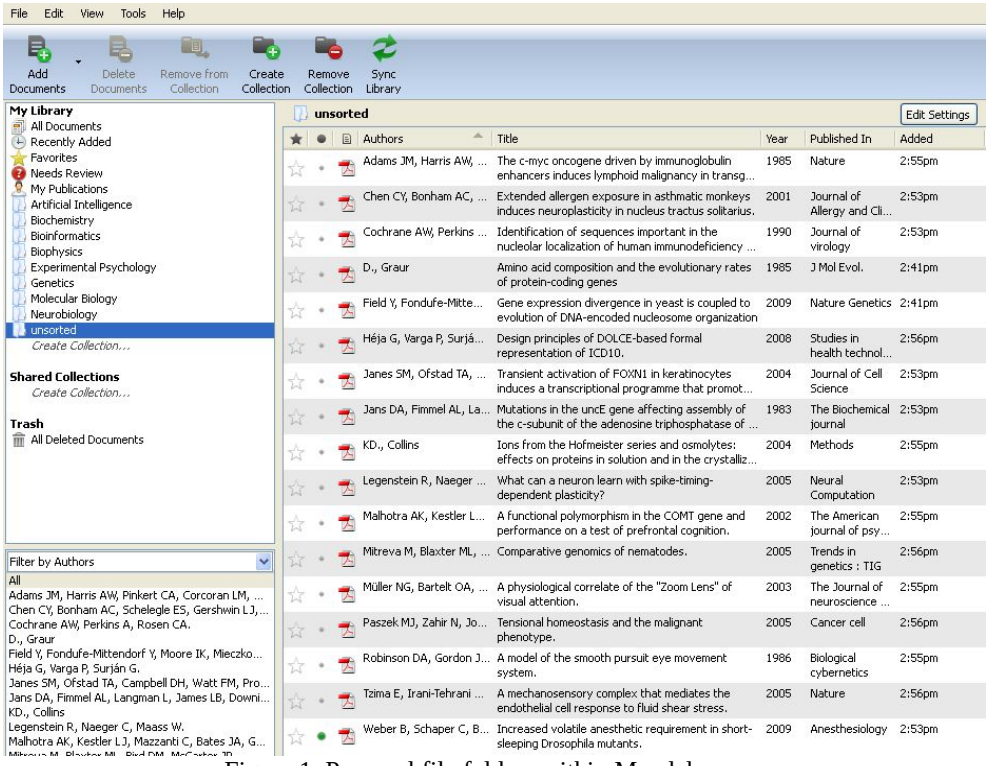

Figure 1: Personal file folders within Mendeley

Our research centers on automatic filing of unseen documents into user-defined categories (in the present context, folders), what can be done to minimize automatic filing errors, and how to keep the folders of manageable size.

[2] Mendeley top level categories correspond to research disciplines, and each is in turn subdivided for finer organization. See http://www.mendeley.com/research-papers/

Questions guiding our work are:

* How can we organize newly-acquired documents automatically into subject categories (of a user's existing folders)?
* What can we do to minimize filing error?
* What options should we present to the user to improve results?
* How can we keep folder sizes convenient for information management?

The paper continues with a discussion of related work. Then system architecture is diagrammed and each of the components is explained. We describe our experiments on two data sets, a data set from Reuters designated as a test collection for text classification research, and a collection of title-abstract excerpts from the biomedical digital library PubMed for a user scenario. We rely upon manual classifications from both data sets as the benchmark for our automatic categories. We discuss strengths and weakness of our system, what can be improved, and future directions for research. Conclusions highlight the significance of our automatic classification and how it will be possible for others to build upon our research.

II. Related work

We are attempting to sort articles into pre-existing categories within a personal information management environment. An early prototype of such an environment was the TEXPROS system of Fan, Liu, and Ng (1999). It extracts an abstract for each document, whether or not the document has its own abstract, and then it organizes these abstracts in folders according to predefined user criteria such as sender, receiver, subject, date, journal or author. Our research focuses on the most complicated of these filing problems: filing according to subject.

We use classification for document organization because it allows the user's familiar category labels to be retained, and because it is more likely to result in coherent groupings with fewer items that do not belong in the group. Automatic classification systems for specific domains have used ontologies to improve indexing (Morais and Ambrósio, 2008). When the document is general or unknown, approaches which use general purpose ontologies such as those created from the online encyclopedia, Wikipedia, might improve classification results to some extent (Syed, Finin and Joshi, 2008). Classification improvements due to adding ontologies, however, probably will be relatively minor (Wang and Domeniconi, 2008).

The procedure begins by transforming words to numbers. We reduce each document to vectors using the Vector Space model (Salton et al, 1975). Alternatively, a document might be reduced to a matrix in a Tensor Space model. But a matrix representation of a document, though it processes more quickly, does not produce appreciably better results (Cai et al, 2006), so we chose the more common Vector Space Model.

Others have experimented with which feature extraction method is best (Rogati and Yang, 2002). Forman (2003) compares 12 feature selection methods. He provides a table with the formula for each feature-selection metric (ibid, p. 1293). Forman points out that different feature selection metrics excel with different data sets, and his appendix presents precision and recall charts that compare metrics according to number of words used for the classification. Liu and Setiono (1995) demonstrated that Chi Square is effective for feature selection. Other research considers which text classification

algorithm performs best (Colas and Brazdil, 2006) and which kernel improves results further (Zhang, Chen and Lee, 2005).

For classification, we use Support Vector Machines (SVM),[3] because SVM has been found to perform well in text mining (Olson and Delen, 2008). An overview of SVM is found in Xue, Yang and Chen (2009). The data, in the case of documents, are composed of words, which the literature, calls attributes, or features, or dimensions. Support vector classifiers use words; other researchers have used word clusters rather than individual words, with inferior results (Bekkerman, El-Yaniv, Tishby, Winter, 2003).

Support Vector Machines belong to the Generalized Linear Models in statistics. In computer science, they are part of Machine Learning, which "learns" based on data that is labeled. A set of SVM algorithms was coded and gathered into a library by Hsu, Chang and Lin (2003). We used a version of the algorithm from their LibSVM toolkit.

SVM classification is an exercise in optimization. The procedure first learns from labeled data where to group the classes, or vectors. Then, with unseen data, it attempts to find maximum separation between vectors. The Support Vector Classifier finds a *hyperplane* – called a classifier – that can divide two classes, or vectors, with maximal separation. Better results are often achieved by relaxing the requirement that the separation be a linear plane. A kernel function may be used to transform the data to non-linear separation. *Soft margin optimization* admits a few classification errors for the sake of better overall classification results. It can be achieved either by adding a soft margin variable, or by using a kernel function to transform the data so it may be separated non-linearly to optimize the groupings. Most people just experiment to see which kernel works best with their particular data set. While the Radial Basis Function (RBF) has been called the "natural choice," no consensus has been reached as to which kernel parameters are optimal for a given data type (Olson and Delen, 2008, p.118, 122). Others have found that after semantically strong features have been extracted, a linear SVM classifier is best (Yilmazel et al, 2008, p. 420).

Computational performance (speed) among methods can differ significantly, which makes some researchers hesitant to assess classification on accuracy alone (Williams et al, 2006). For our purposes, speed is not critical since the computations can be set up to execute while the user's attention is elsewhere.

In personal information management, automatic document classification might create categories or folders that contain too many documents for a user to manage constructively. So we use a clustering algorithm to subdivide large categories. The problem is then that the clusters are unlabelled. Maqbool and Babri (2006) created a cluster-labeling algorithm that uses frequency and inverse frequency to choose terms to be used as labels. Tseng (2010) presents a generic method to label the clusters which first extracts category-specific terms using correlation coefficient algorithm (the square root of the chi-square) and then maps these to WordNet terms. Our method to label

[3] Support Vector Machines (SVM) were developed by Vapnik in the 1990s (Vapnik, 1995). Of all the classification methods, Support Vector Machines have been found to perform with high accuracy in many high dimensional applications, including medical diagnostics and bioinformatics, face recognition, as well as text mining (Olson and Delen, 2008, 111-123).

clusters is similar to Tseng's except that we use the Chi Square statistic. Chi square turns out to be the square of the correlation coefficient (Tseng, 2010, p. 2248).

Overly large categories are managed by clustering. Clustering puts unlabelled articles into coherent groupings, and in this case, it would subdivide categories. This is done in the Vivísimo search engine, for example.[4] The problem is that the new labels that will be applied could be unfamiliar to the user. Moreover, the labels designated for one cluster might be a subset of another label. For example, one category might be grain, and another, corn and another, rice. The corn cluster and the rice cluster logically are subsets of the grain, and so all could be placed into a single category. We adjust for this circumstance and help the user get a quick overview of contents by attaching several labels to each newly-made cluster.

Cross validation is often used to predict how well the classifier would perform on unseen data based on performance on labeled data. In *z*-fold cross validation, we divide the training set into *z* equal-size subsets. Each subset slice is tested sequentially using the classifier trained on *z*-1 slices. Thus, each subset is predicted once. Cross validation accuracy is the percentage of data that is classified correctly.

Additional evaluation measures are precision and recall, and their combination in the harmonic mean, called the F measure.

$$P = \frac{TruePositive}{TruePositive + FalsePositive} = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ items\ retrieved}$$

$$R = \frac{TruePositive}{TruePositive + TrueNegative} = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ relevant\ items\ in\ a\ collection}$$

$$F\ measure\ (harmonic\ mean\ of\ precision\ and\ recall) = \frac{2PR}{P+R}$$

A true positive is an item that is correctly assigned to a category; a false positive is an item assigned that should not be; a true negative is an item that should not be assigned, and is not assigned. False and true positives (incorrect and correct assignments of items to categories) may be visualized in what is called a confusion matrix. An ROC graph captures all the points in the confusion matrix but is somewhat difficult to interpret. We have translated results out of a confusion matrix to simplify the presentation.

III. Method

The focus of our current research is on an actual use case. For this, we worked with personal digital library software Mendeley, and a selection of article extracts from the biomedical digital library PubMed that had been labeled with Mendeley subject categories. Our particular method, using Chi square for feature extraction and SVM for the classification, has been employed by others (Zhang and Zhang, 2008).

[4] http://vivisimo.com, Retrieved May 13, 2010

An overview of our procedure is shown in Figure 2. In brief, we pre-process the documents first. This consists of removing stop words by comparison to a standard list. We make capital and lower case letters uniform by converting all to lower case, although this is not essential for classification. Then suffixes were cropped in what is known as stemming or lemmatization. For example, swimming and swims would be stemmed to swim. The effect of stemming is that similar words get counted as repetitions of the same word, which results in a lower dimension, simpler Space Vector Model.

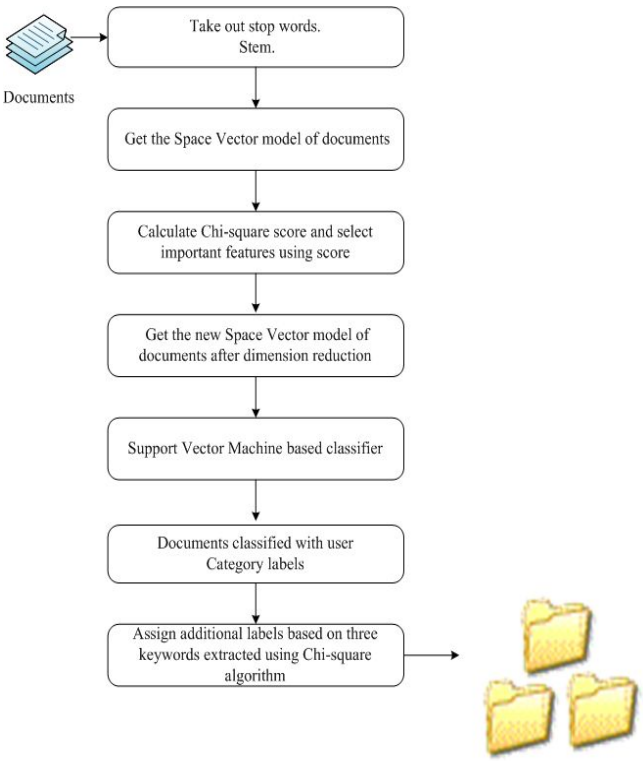

Figure 2: Diagram of experimental procedure for classification.

The Vector Space Model extracts content-bearing words to represent each document as a numerical vector. It is a "Bag of Words" approach. Word scores are computed based on semantic significance of the word in the document and in the collection as a whole. For example, the word "house" that appears once in an article on medical equipment is not as good an indicator of category topic as "microscope". Since each word represents a dimension, the dimensionality of a document may be high. A thousand-word document would have a dimensionality of 1000.

We use tf–idf to compute a score for each word by [dividing the number of occurrence of the word by the total number of words in the particular document (tf)] and multiplying by the quotient of [total number of documents by the number of documents containing the term (idf)]. The Chi Square statistic compares the score of the word to the score of the category for all words in the document, and it ranks words according to semantic significance. When the same word is found in more than one category, word scores are compared and only the single instance of the word that scores the highest is retained.

Once we have isolated words for each category using Chi Square, we are ready to create classifiers that will use these words to group unseen documents into the category. For this, we use the Support Vector Classifier (SVC) algorithm. A classifier uses as input the tf-idf numbers of documents along with the category number and, along with some additional computations, creates a hyperplane to separate the documents into two categories. Each document is either assigned to the classifier's category (positive), or rejected from the classifier category (negative). A negative assignment diverts the document so that it is run in another classifier, with the possibility of being assigned to another category.

We used the WEKA data mining tool set developed at the University of Waikato in New Zealand.[5] The version of the SVC algorithm we used is not native to WEKA,; for this, we downloaded separately the LibSVM plugin.[6] If our project were to be implemented on a server, we would recommend LibSVM because WEKA classifiers admits only small data sets.

We found through experimentation that the Reuters news data classified most accurately with a linear kernel, whereas the PubMed articles classified most accurately with the RBF kernel (Figure 6). This distinction seems largely data-dependent. We could program a system to select the optimal performance kernel based on the articles the user had organized into category folders, and we could test the model via cross validation. This is expanded in the Discussion section below.

In cases where the classification algorithm assigns overly many documents to a folder so that the folder is no longer convenient for personal information management, we propose subdividing the folder by clustering. Some have offered users the opportunity to customize clustering results, as does Bekkerman et al. (2007) and Huang and Mitchell (2006), but the customization entails that the user present examples of what he wishes clustered. For our part, we feel that mixed initiative clustering will be too much of a chore for users, so we wholly automate the clustering.

IV. Experiments

Data

Subjecting different data sets to the same procedure will yield different accuracies, so we used more than one data set to test our procedure. We use the Reuters news article collection designated 21578 that appeared on the Reuters news line in 1987. The articles were assembled and indexed by personnel from Reuters Ltd. and the Carnegie Group, Inc.[7] The entire corpus contains 11,367 manually-labeled documents classified into 82 category groups.

[5] WEKA may be downloaded as of June 2010 from http://www.cs.waikato.ac.nz/ml/weka/

[6] LIBSVM: A Library for Support Vector Machines, as of July 7, 2010 at http://www.csie.ntu.edu/tw/~cjlin/libsvm

[7] As of April 2010, the corpus was at http://www.daviddlewis.com/ resources/testcollections/reuters21578/

Our second corpus consists of 1000 title/abstract document excerpts downloaded from the biomedical digital library PubMed, and that has been classified into 516 categories.

Procedure

Sampling method. We selected in each data set the 10 categories that contained the largest number of articles to help ensure that we would have enough data (that is, articles) to train the category classifier. This amounted to 500 articles from Reuters, and 308 abstracts from the PubMed set.

Pre-processing. We prepared the data by converting all text into lower case, removing stop words,[8] and applying a stemming algorithm.

Processing. We used the Space Vector Model based on tf-idf weighting to correlate frequency of word in each document with that in the document set as a whole. This produces a numerical score for each word. We used the Chi Square statistic in an algorithm that uses the word scores to rank the features within the category. The result is that the higher numbered words "belong" more to the category and are more important indicators of category. These words become input for the SVM algorithm, which creates a classifier for each category. Each category has learned a different instance of the classifier.

Cross-validation. The benchmark for accuracy is the manual classification of the articles, with accuracy defined as the number of articles automatically classified into the manually-determined categories. The document set is divided into parts and the model is tested on unseen data. In this case, we divided the collection into 10 parts, using 9 for training and the remaining 1 for testing. We re-shuffled the parts and used a different 9 for training and the remaining part for testing, over the course of 10 runs. The results from the 10 runs are then averaged. We achieved 96% accuracy on the Reuters data set, and 89% accuracy on the PubMed data set. (see Figure 3).

Assigning additional labels

After each document is assigned a category, we run words from each category through the Chi Square algorithm to find the three highest scoring words that appear in the document cluster. We can then assign additional words as labels to clarify category folder contents.

V. Results

Reuters data set reduced to 100 features		
Correctly classified instances	482	96%
Incorrectly classified instances	18	4%
Total number of documents	500	
Total features (words)	4256	
PubMed data set reduced to 100 features		
Correctly classified instances	274	89%
Incorrectly classified instances	34	11%

8 The stop word list we used was downloaded from the *Journal of Machine Learning Research*, retrieved April 30, 2010 from http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

Total number of documents	308
Total features (words)	4799

Figure 3: Classification results for the two data sets, with 100 words (that is, features) for each.

Figure 3 shows resulting classification accuracy for the two data sets under parallel constraints (here, the same number of features run through the classifier and the same kernel).

Figure 4 compares the F measure results per number of features in the two data sets. Both data sets peak at roughly 200 features. This suggests that it might be unnecessary to run a large numbers of features for optimal results on an unseen data set. How many documents are needed to achieve 200 features per category? This depends upon the total number of categories as well as the average length of the documents.

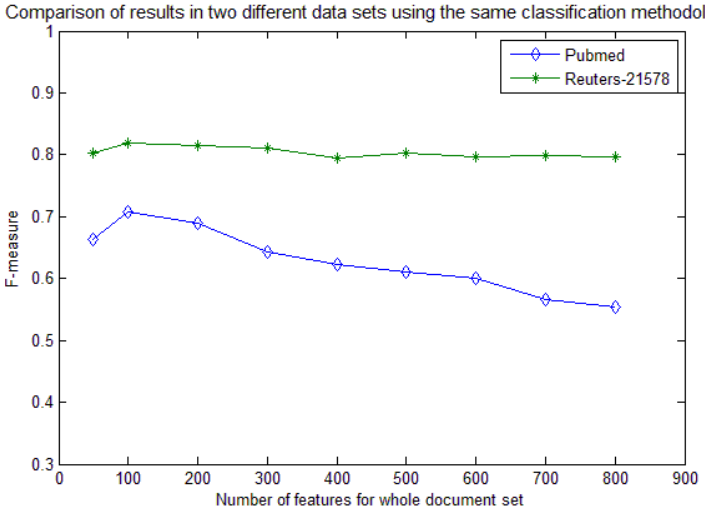

Figure 4 Accuracy measures from same classification procedure in two different data sets suggest that 100-200 features (words) suffice.

VI. Analysis

Why did the classifiers perform better on the Reuters data? The Reuters data have been called simple to classify (Bekkerman and Allan, 2003). In the category "ship" randomly sampled from this data set, quite a few instances of the word "ship" are found among key features extracted (Figure 5a).

The comparison between the Reuters and PubMed data in Figures 5a and 5b shows the top Chi Square words of documents assigned to the Reuters category "ship" and PubMed category "neurobiology." The features that do not appear to be words are actually abbreviations. FVN is a Netherlands insurance company in the Reuters data sets, and NMDA in the PubMed data set is a glutamate receptor.

In the given samples, document 4 in the Reuters data set (Figure 5a) and documents 19, 24, 28, 30 in the PubMed data (Figure 5b) have been mis-assigned to their respective categories. The reason? Guilt by association. The features in the mis-classified

documents are associated with other documents that are correctly classified, but in these particular cases, the association misleads into a classification assignment that is false.

Category: SHIP (From Reuters-21578)	
Document 1:	vessel, ship, herald, ferry, rotterdam, transport, enterprise, cargo, capsize, zeebrugge, channel, british, disaster, crew, owner, navigate, harbour, water, accident, load, belgian, free
Document 2:	vessel, dispute, transport, water, northwest, ice
Document 3:	vessel, ship, port, halt, city, northwest
Document 4:	ship, port, cargo, channel, handle
Document 5:	vessel, ship, herald, ferry, enterprise, cargo, passenger, zeebrugge, shipowner, townsend, disaster, court, crew, navigate, accident, free
Document 6:	vessel, ship, port, strike, transport, seamen, marine, janeiro, harbour, halt, de, merchant, idle
Document 7:	ship, port, cargo, handle
Document 8:	port, strike, dispute, rotterdam, transport, cargo, fnv, redundancy, de, louw
Document 9:	port, dispute, rotterdam, transport, cargo, fnv, court, redundancy
Document 10:	herald, ferry, transport, enterprise, passenger, dead, townsend, channel, british, disaster, rescue, owner, water, de, belgian, hundred, free, ice

Figure 5a: Key words extracted per document and ranked via Chi Square from the Reuters category "Ship". Highlighted document 4 is a mis-classification.

Category: NEUROBIOLOGY (From PubMed)	
Document 17:	neuron, depolarization, potential, amplitude, record, fire, action, afferent, serotonin, impulse, ht, accompany, fiber, mechanical
Document 18:	neuron, synapse, hippocampus, efficacy
Document 19:	neuron, synapse, postsynaptic, efficacy, afferent, input
Document 20:	neuron, synapse, brain, potential, underlie, efficacy, blockade, gene, mechanical
Document 21:	synapse, brain, plastic, underlie, accompany, cerebellar, gene
Document 22:	neuron, synapse, hippocampus, amplitude, plastic, NMDA, spine, accompany, mechanical
Document 23:	axon, record, stimulate, cardiovascular, fire, impulse, baroreceptor, discharge, nerve, blockade, reflex, artery
Document 24:	neuron, fire, input
Document 25:	neuron, brain, evoke, depolarization, potential, record, plastic, nt, tractus, action, afferent, respiratory, clamp, solitary, input, nerve, fiber, central, mechanical
Document 26:	synapse, brain, plastic, underlie, efficacy, mechanical
Document 27:	neuron, synapse, brain, evoke, amplitude, postsynaptic, inhibitory, NMDA, latency, cerebellar, mechanical
Document 28:	neuron, synapse, synapse, postsynaptic, record, afferent, input
Document 29:	synapse, evoke, depolarization, potential, synapse, postsynaptic, record, stimulate, presynaptic, action, inhibitory, central, mechanical, ca
Document 30:	underlie, reflex, central, mechanical

Document 31:	neuron, synapse, synapse, plastic, clamp, input

Figure 5b: Highlighted documents are mis-classifications. In this data set as in the Reuters examples in Figure 5a, at the key feature words in the mis-classified documents appear in the correctly classified documents.

Our experiments here have shown that excellent classification results are possible with title and abstract only. Would we have achieved better results had we used the full text of the PubMed articles along with title and abstract? Further experiments are needed.

VII. Discussion

What makes a good classifier?

We used only a subset of the Reuters 21578 corpus. For the sample we used, the "Copper" classifier works best, achieving 98% accuracy, while the worst is "Grain," achieving only 59% accuracy. For the PubMed data set, the best classifier is "Experimental Psychology" at 71%, while the worst is "Microbiology" at 0% accuracy.

Our best classifiers did not necessarily become more accurate when built through a larger training set. However, the classifiers that performed the worst had among the fewest documents in the training set, as shown in Figure 6. The best classifiers include a good distribution of words as determined by tf-idf weighting that are strong indicators of category. Figure 6 shows that the Reuters data performed best with the linear kernel, whereas the PubMed data performed best with the RBF kernel. We do not know why, but it might have to do with the particular terms and their distribution in the document.

Reuters 21578

Classifier	Documents assigned manually to categories	Binary SVM	
		Number of documents assigned automatically to the correct categories: linear kernel	Number of documents assigned automatically to the correct categories: RBF
Ship	46	35	41
Jobs	49	44	28
Sugar	66	60	66
Grain	28	19	24
Crude	105	94	98
Money	52	45	25
Trade	44	35	38
Coffee	44	42	40
Copper	42	42	42
Cotton	24	21	23

PubMed

Classifier	Documents assigned manually to categories	Binary SVM	
		Number of documents assigned automatically to the correct categories: linear kernel	Number of documents assigned automatically to the correct categories: RBF
Bioinformatics	60	35	41
Genetics	26	5	18
Neurobiology	57	25	40
Biochemistry	28	9	24
Molecular Biology	29	9	25
Cellular Biology	23	1	17
Experimental Psychology	34	14	27
Artificial Intelligence	23	6	18
Microbiology	8	0	7
Biophysics	20	5	16

Figure 6: Classifier categories and number of training documents per category, with comparison of the linear and RBF kernel.

How could the procedure be modified to make fewer classification mistakes?

A possible way to improve classification for the PubMed data would be to bridge the gap between terms found in the biomedical articles and the category terms of Mendeley. In similar document classification experiments, Wang and Domeniconi (2008) and Gabrilovich and Markovitch (2006) extracted concepts from Wikipedia, along with synonymous and associated concepts, and used these to enrich the features extracted from the document set. Even though the Gabrilovich and Markovitch results improved with the ontology-enrichment, the improvement was only marginal (Wang and Domeniconi, 2008, p. 20, Table 10).

Another method to improve results in both data sets would be to weight words in article titles more than words in the abstract, a method that appeared in Dumais et al. (1998). Another direction of research might be to use phrases as well as words for classification. It has also been noted that bi-grams (phrases) may be more predictive than single words (Yang and Pedersen, 1997).

Our classification method permits recall to suffer for the sake of precision. That is, we permit the cost of errors in recall to be less than the cost of errors in precision. In some applications, such as in threat research, a false negative could lead to risky consequences

if a critical problem goes undetected. For the goal of increasing recall, then, other classification methods might be preferred (Seo and Sycara, 2008).

What options should we present to the user?

We require the user's existing documents in labeled folders, say, in Mendeley information management software, as preparation for automatic filing. The procedure might begin with a "Test for Accuracy" button, which would start an algorithm to compare results with different SVM kernels to see which would yield highest classification accuracy. The system then would present the user with a message such as "Expect 75% accuracy or greater when adding documents to the current folders," with an option for more information that would show the number of features, kernel selected, and technical details.

The system requires a minimum accuracy for placing unseen documents in folders for it to be a reliable desktop tool. What percent accuracy is required in document organization should be tested through user study. For the sake of discussion, let's say the system's accuracy tolerance is set to 70%. That would mean that when the system self-tests and obtains an accuracy of 69% or less, and some user interaction to improve accuracy would be requested. The system would either ask the user for more documents to re-test, or it would ask the user to merge the contents of two folders. In re-testing a classification, the system could either send each item in a category to each classifier and then compare F scores of each classifier to find an alternative category for filing, or else switch to multi-class SVM to determine an alternate category. A more involved approach would be to retain inferior classifications and ask the user to select those items that do not fit with the others, so that the system could learn it had made an error. This is the protocol suggested for the Lighthouse system by Leuski and Allen (2000). This possibility requires some effort on the part of the user and so we do not believe many users will find it attractive.

Rather than ask the user to make (negative) decisions about what may have been put in a category wrongly, the system might ask the user to make (positive) classification decisions and add documents to the category that did not self-classify accurately. A message might be "Please add [number needed to make 10 total] documents to the [folder achieving low accuracy]. Or perhaps, the system could adopt Google's humility. "Well, this is embarrassing. We do not have enough documents in the [folder achieving low accuracy] to perform the classification. However, if you add more documents we would do better." So even a positive decision requires some effort, so we think many users will find this option unattractive too.

Perhaps the best approach from a usability standpoint would be to ask the user his opinion whether two categories could be merged. The message could be something like "Consider merging the [folder achieving low accuracy in classification] with [second best folder category]. For example, a message might be "Consider merging the Microbiology folder with Bioinformatics."

Should we limit the number of documents per category folder?

Our sample data sets include between 8 and 105 documents per category (Figure 7a and 7b). Perhaps 15 or 20 are minimally necessary per category to train the classifier. But do users care to look over so many documents to find the one of interest? A user study would be necessary to confirm what is preferred, but we believe that 70 documents is too

large for a single folder, and 5 may be so small that it just wastes space. We propose for the purposes of this present study that between perhaps 6 and 40 documents per folder is manageable.

Folder size (or category size) thus becomes a factor in the personal information environment. So we run a clustering algorithm over overly-large folders to organize the documents into subgroups, or sub-folders.[9] We use K-means clustering which has been found to work well independent of the data domain.

As a default for subdividing large folders, we create 3 subgroups. After clustering, if one of the three subgroups has more than 40 documents, we will re-classify the documents and create 4 subgroups instead (changing the default from 3 to 4). But if after the initial clustering, one of the 3 subgroups has 5 or fewer documents, we change the default from 3 to 2 and re-run the clustering to make 2 subgroups.

It is necessary to label the newly-formed grouped so that users have an idea of what the folder contains. We create category labels automatically by running Chi Square to assign word scores to document words. We can then use these words to label the subgroups, or subfolders. One label is probably insufficient, and anyway might be ambiguous, so we assign 3 to 5 labels. In the event that the same word appears in more than one subgroup, we can use these scores also to select the highest-scoring word and remove duplicates so that the same word is not used as a label for more than one subfolder.

Reuters 21578

Classification category	**Documents assigned manually to categories**	**Becomes subcategories**	
		Number of documents in subgroup	**Keywords for subgroup**
Ship	46	18	Free, Ferry, Enterprise, Herald, Channel
		17	Strike, Union, Port, Dispute, Marine
		11	Line, Northwest, Snow, Sea, Rotterdam
Jobs	49	16	Record, Worst, Compile, Number, Unadjusted
		20	Workforce, Office, Unemployed, Job, Manufacture
		14	Labor, Insure, Program, Prior, Regular
Sugar	66	36	Beet, Yield, Hungary, Hectare, Sow
		9	Intervene, Rebate, Commis, License, Maximum
		21	West, Germany, European, France, French
Crude	105	11	Distil, Refinery, Gasoline, PDVSA, Price
		30	Crude, Explore, Drill, Recommend, Study

[9] We use sub-folders and subgroups interchangeably.

		14	Earthquake, Ecuador, Jungle, Balao, Repair
		26	Saudi, Arabia, Ceil, Defend, Lift
		18	Bbl, Post, Raise, WTI, Intermediate
Money	52	17	Fed, Business, March, Money, Loan
		23	Supply, Growth, Define, Asset, Exclude
		12	Dlr, Lend, Week, Bundesbank, Sterling
Trade	44	11	Japanese, Surplus, Japan, Electron, Baker
		19	Bill, Subcommittee, Toughen, Unfair, Democrat
		14	Billion, Deficit, Import, Retaliate, Legislate
Coffee	44	16	Gaviria, Cesar, Factor, Gilberto, Arango
		14	Colombia, Delegable, Intern, Consume, Talk
		14	Delegable, Council, Group, Meet, April
Copper	42	18	Lower, Effect, Cathod, Immediate, Lb
		7	Fire, Noranda, Kill, Trap, Full
		17	Chile, Confirm, Zambia, Chilean, Cent

Figure 7a The table shows the number of Reuters documents manually assigned to the Reuters categories, and for each sub-grouping, 5 keywords found via the Chi Square statistic which we use for folder labels.

Pubmed

Classifier	**Documents assigned manually to categories**	**Becomes subcategories**	
		Number of documents in subgroup	**Keywords for subgroup**
Bioinformatics	60	17	Genome, Biology, Analyze, Chain, Indelible
		25	Gene, Regulatory, Genetic, Biologist, Web
		18	Thermophile, Temperature, Residual, Acid, Amino
Genetics	57	21	Record, Response, Afferent, Nerve, Nucleus
		18	Dendrite, Spine, IPSC, Acute, Axon
		17	Memory, Neural, Depend, Learn, Gene

Figure 7b The table shows the number of PubMed documents manually assigned to the Mendeley categories, and for each sub-grouping, 5 keywords found via the Chi Square statistic which we use to label the newly-formed subfolders.

VIII. Summary

Our contribution rests in the proposed adaption of standard classification and labeling procedures to personal information management. We use as training data the articles the user has already filed and the (presumably) subject label he has already assigned to each category folder for the machine learning task. We ran experiments on two data sets to see whether our accuracy was domain dependent, and found that our procedures worked well on both more general data and also domain-specific, technical medical data, although procedures worked better in the more general domain.

The general data was the Reuters text categorization set 21578 and the domain-specific data was a set of article titles with abstracts from PubMed. Both had manual classifications that we used to measure automatic classifications. The Chi Square feature selection method to isolate keywords to classify, followed by the Support Vector Machine algorithm to classify, yielded 96% accuracy on the "easy" Reuters data and 89% accuracy on the more technical PubMed data. The PubMed abstracts were part of our use-case scenario, demonstrating a research article collection stored in the personal information management software Mendeley. We proposed a procedure to test classifications without inconveniencing the user, and some choices to present to the user to improve classification accuracy for unseen documents.

Acknowledgment

We are grateful for several Skype discussions with Mendeley director, Dr. Jason Hoyt, who suggested the research problem and provided the PubMed data with Mendeley category classifications that we used for experiments.

References

Barreau, D. and Nardi, B. A. (1995). Finding and reminding: file organization from the desktop. *ACM SIGCHI Bulletin 27*(3), 39-43.

Bekkerman, R. and Allan, J. (2003) Using Bigrams in Text Categorization. *CIIR Technical Report IR-408*, 1-10.

Bekkerman, R., El-Yaniv, R., Tishby, N. Winter (2003). Distributional Word Clusters vs. Words for Text Categorization *Journal of Machine Learning Research 3*, 1183-1208.

Bekkerman, R., Raghavan, H., Allan, J. Eguchi, K. (2007). Interactive clustering of text collections according to a user-specified criterion. *Proceedings of the 20th International Joint Conference on Artificial Intelligence. Hyderabad, India*, San Francisco, CA: The International Joint Conferences on Artificial Intelligence, Inc., 684-689.

Cai, D., He, X, and Han, J. (2006). Tensor space model for document analysis. *SIGIR '06, August 6-11, Seattle, Washington, USA*, 625-626.

Colas, F. and Brazdil, P. (2006) Comparison of SVM and some other classification algorithms in text classification tasks. In *Artificial Intelligence in Theory and Practice* (pp. 169-178). Boston: Springer.

Dumais, S., Platt, J., Heckerman, D. and Sahami, M. (1998). Inductive Learning Algorithms and Representations for Text Categorization. In *Proceedings of the 17th International Conference on Information and Knowledge Management*, 148-155.

Fan X., Liu Q.H., Ng P. (1999). An automated document filing system. *Journal of Systems Integration, 9,* 223-262.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research 3*, 1289-1305.

Gabrilovich, E. and Markovitch, S. (2006). Overcoming the brittleness bottleneck using Wikipedia: enhancing text categorization with encyclopedia knowledge. *Proceedings of the 21st National Conference on Artificial Intelligence*- Volume 2 Boston, Massachusetts, 1301-1306.

Grira, N., Crucianu, M, Boujemaa (2005). Unsupervised and semi-supervised clustering: a brief survey. In *A Review of Machine Learning Techniques for Processing Multimedia Content, Report of the MUSCLE European Network of Excellence (6th Framework Programme*), 1-12.

Guinepain, S. and Le Gruenwald (2005). Research issues in automatic database clustering. *SIMOD Record, 34 (1),* 33-38.

Henderson, S. (2005). Genre, Task, Topic and Time: Facets of Personal Digital Document Management. *CHINZ '05, July 6-8, Auckland, NZ*, 75-82

Hsu C. W., Chang C.C., Lin C.J. (2010). A practical guide to support vector classification. *Technical Report,* Department of Computer Science & Information Engineering, National Taiwan University, Taiwan. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

Huang, Y. and Mitchell, T. M. (2006). Text clustering with extended user feedback. *SIGIR '06, August 6-11, 2006, Seattle, Washington, USA*, 413-420.

ICT 2010. "Me and my files" ICT 2010. Retrieved May 2, 2010 from http://cordis.europa.eu/ictresults/index.cfm?section=news&tpl=article&BrowsingType=Features&ID=91255

Leuski, A. and Allan, J. (2000). Lighthouse: showing the way to relevant information. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis'00)*, Salt Lake City, Utah, USA, October 9-10, 2000. IEEE Computer Society, 125-130.

Liu H. and Setiono R. (1995) Chi2: Feature selection and discretization of numeric attributes. *Proceedings of the IEEE 7th International Conference on Tools with Artificial Intelligence, 5-8 Nov 1995, Herndon, Virginia, USA,* 388-391

Magbool Q. and Babri H.A. (2006). Automated software clustering: An insight using cluster labels. *The journal of Systems and Software 79,* 1632-1648.

Morais, E. A.M. and Ambrósio, A.P.L. (2008). Automatic domain classification of jurisprudence documents. *Proceedings of the 2008 Euro American Conference on Telematics and Information Systems (EATIS) '08, September 10-12, Aracaju, Brazil* [6 pages].

Olson, D. L. and Delen, D. (2008). Support Vector Machines. *Advanced Data Mining Techniques* Berlin: Springer.

Rogati, M. and Yang, Y. (2002). High performing feature selection for text classification. *CIKM '02, November 4-9, 2002, McLean, Virginia,* 659-661.

Salton, G., Wong, A. and Yang, C.S. (1975). A vector space model for information retrieval. *Communications of the ACM*, 18 (11): 613–620.

Seo, Y-W and Sycara, K. (2008). Addressing insider threat through cost-sensitive document classification. In H. Chen, E. Reid, J. Sinai, A. Silke and B. Ganor (Eds.) *Terrorism Informatics: Knowledge Management and Data Mining for Homeland Security*. (pp. 451-472). New York: Springer.

Syed, Z.S., Finin, T. and Joshi, A. (2008). Wikipedia as an ontology for describing documents. *International Conference on Weblogs and Social Media, March 30-April 2, 2008, Seattle Washington, U.S.A.*, 136-144.

Tseng, Y.H. (2010). Genetic title labeling for clustered documents. *Expert Systems with Applications, 37*, 2247-2254.

Vapnik, V. (1995)[1998]. *The nature of statistical learning theory*. Second edition Springer.

Wang, P. and Domeniconi, C. (2008) Building semantic kernels for text classification using Wikipedia. *KDD, August 24-27, 2008, Las Vegas, Nevada,* 713-721.

Williams, N., Zander, S., Armitage, G. (2006) A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification, *ACM SIGCOMM Computer Communication Review 36* (5), 5-16.

Xue, H. Yang, Q. and Chen, S. (2009). SVM: Support Vector Machines. In X. Wu and V. Kumar (Eds). *The top ten algorithms in data mining* (pp. 37-59). Boca Raton: CRC Press.

Yang, Y. and Pedersen, J. O (1997). A Comparative Study on Feature Selection in Text Categorization. Ed. D. H. Fischer, *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, July 8-12*, 412-420.

Yilmazel, O, Symonenko, S. Balasubramanian, N. and Liddy, E. (2008). Leveraging one-class SVM and semantic analysis to detect anomalous content. In H. Chen, E. Reid, J. Sinai, A. Silke and B. Ganor (Eds.) *Terrorism Informatics: Knowledge Management and Data Mining for Homeland Security*. (pp. 409-424). New York: Springer.

Zhang, M. and Zhang, D. (2008). Trained SVMs <sic> based rules extraction method for text classification. *IEEE International Symposium on IT in Medicine and Education, 12-14 Dec. 2008, Xiamen, China*, 16-19.

Zhang, D. Chen, X. and Lee, W. S. (2005) Text classification with kernels on the multinomial manifold. *SIGIR '05, August 15-19, Salvador, Brazil*, 266-273.