Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

# Developing an automatic document classification system

*A review of current literature and future directions*

J. David Brown

**Defence R&D Canada – Ottawa**

Canada

Technical Memorandum
DRDC Ottawa TM 2009-269
January 2010

# Developing an automatic document classification system

*A review of current literature and future directions*

J. David Brown
Defence R&D Canada – Ottawa

**Defence R&D Canada – Ottawa**
Technical Memorandum
DRDC Ottawa TM 2009-269
January 2010

Principal Author

*Original signed by J. David Brown*

J. David Brown

Approved by

*Original signed by Julie Lefebvre*

Julie Lefebvre
Head/NIO Section

Approved for release by

*Original signed by Brian Eatock*

Brian Eatock
Head/Document Review Panel

# Abstract

Assigning a security classification to a document is typically a labour-intensive manual process performed by a trained professional who must read and understand the document and subsequently apply the rules of an organization's security policy. Automating this process would increase organizational efficiency and would nicely complement newly proposed data-centric security systems where all data must be accurately labelled with the appropriate classification. This paper introduces some of the challenges faced in developing an automated security classification system and discusses current text categorization technologies (dimensionality reduction and machine learning techniques) which would be the key enablers of such a system. In addition to the technology review, several avenues of research are proposed to evaluate a number of potential solutions to the security classification problem.

# Résumé

Attribuer une classification de sécurité à un document est généralement un processus manuel exigeant en main-d'oeuvre et exécuté par un professionnel chevronné qui doit lire et veiller à bien comprendre le document, et par la suite appliquer les règlements de la politique sur la sécurité de l'organisation concernée. Le fait d'automatiser ce processus permettrait d'augmenter l'efficacité de l'organisation, en plus de compléter adéquatement les systèmes de sécurité axés sur les données récemment proposées, dans lesquels les données doivent afficher la classification appropriée. Ce document ci présente quelques uns des défis confrontés au moment d'élaborer un système de classification de sécurité automatisé, et aborde les technologies actuelles de catégorisation des textes (techniques de réduction de la dimensionnalité et d'apprentissage automatique), qui constituent les principaux outils habilitants d'un tel système. En plus de l'examen de la technologie, plusieurs méthodes de recherche sont ici proposées pour évaluer un grand nombre de solutions possibles au problème de la classification de sécurité.

This page intentionally left blank.

# Executive summary

## Developing an automatic document classification system

J. David Brown; DRDC Ottawa TM 2009-269; Defence R&D Canada – Ottawa; January 2010.

**Background:** Current research in information management security has identified a need for accurate and efficient methods to classify sensitive government and corporate information, including newly-created data and unlabelled legacy data residing in an organization's electronic archives. A common process for assigning a security classification to a potentially sensitive document is to have a trained professional read the document and decide on the classification based on the organization's security policy. This process can be time consuming and can lead to inconsistency depending upon the training and background knowledge of the evaluator. Automating the process of security classification would not only increase organizational efficiency but would facilitate the integration of newly-proposed data-centric security models [1] into the organization's IT infrastructure.

**Principal results:** This paper presents a comprehensive survey of existing technologies for text categorization, focusing primarily on statistical natural language processing techniques. Pre-processing of the text using stemming and tokenization, reducing the dimensionality of the text through feature selection and/or feature extraction, and ultimately categorizing the text using a machine learning algorithm are all described in detail. State of the art research in sentiment analysis and spam filtering are also discussed. In addition, this paper identifies several key challenges that distinguish the task of assigning a security classification from typical text categorization problems; these challenges include logistical issues such as the difficulty of obtaining a large enough set of "test" documents on which to conduct research, and more technical issues related to the fact that most text categorization research focuses on sorting documents according to topic (not according to sensitivity).

**Significance of results:** By identifying the pertinent research and the existing technologies that would facilitate the development of a classification system, this paper lays the foundation for a future research program focused on automating the process of security classification.

**Future work:** A number of open problems and directions for future work are presented in Section 4. In order to conduct initial research, the paper proposes using "test" documents from a vast repository of recently declassified documents found

at the Digital National Security Archive [2]. An initial open problem that must be addressed is determining the performance of the proposed state of the art text categorization techniques when applied to the task of security classification. Research is also required to determine how well these techniques will generalize for large collections of data on different topics from disparate parts of the organization. Finally, by carefully studying how an automated system identifies sensitive data, some progress may be made towards finding a solution to the known problem of data aggregation.

# Sommaire

## Developing an automatic document classification system

J. David Brown ; DRDC Ottawa TM 2009-269 ; R & D pour la défense Canada – Ottawa ; janvier 2010.

**Contexte :** Les recherches réalisées actuellement dans le domaine de la sécurité de la gestion de l'information révèlent le besoin de méthodes précises et efficaces pour classifier l'information gouvernementale et organisationnelle de nature délicate, notamment la création de nouvelles données et l'intégration de données existantes sans référence dans les archives électroniques de l'organisation. Le processus général pour attribuer une classification de sécurité à un document de nature potentiellement délicate consiste à demander à un professionnel chevronné de lire le document et de déterminer la classification en fonction de la politique sur la sécurité de l'organisation. Ce processus peut exiger beaucoup de temps et entraîner une incohérence, selon la formation et les connaissances préalables de l'évaluateur. En plus d'accroître l'efficacité de l'organisation, le fait d'automatiser le processus de classification de sécurité permettrait de faciliter l'intégration des modèles de sécurité axés sur les données récemment proposés [1] dans l'infrastructure de TI de l'organisation.

**Principaux résultats :** Ce document présente une enquête exhaustive sur les technologies existantes de catégorisation de textes et porte principalement sur les techniques de traitement du langage statistique naturel. On y explique en détail le prétraitement du texte à l'aide de l'indexation par radicaux et de la segmentation en unités, la réduction de la dimensionnalité du texte par la sélection et/ou l'extraction de fonctions, et finalement la catégorisation du texte à l'aide d'un algorithme d'apprentissage automatique. La recherche de pointe dans l'analyse de sentiment et le filtrage antipourriel fait également l'objet de discussions. En outre, ce document relève plusieurs défis importants qui distinguent la tâche d'assigner une classification de sécurité des problèmes typiques de catégorisation de texte ; ces défis comprennent des problèmes logistiques comme la difficulté d'obtenir un ensemble assez vaste de documents ≪d'essai≫ avec lesquels réaliser la recherche et des difficultés plus techniques liées au fait que la majorité de la recherche sur la catégorisation porte sur le tri des documents selon le sujet (et non selon la sensibilité).

**Portée :** En relevant les recherches pertinentes et les technologies existantes qui faciliteraient le développement d'un système de classification, ce document jette les assises d'un futur programme de recherche sur l'automatisation du processus de classification de sécurité.

**Recherches futures :** Certains problèmes en suspens et certaines pistes de recherches à venir sont présentés à la section 4. Afin d'effectuer la recherche initiale, le document propose d'utiliser des documents ≪d'essai≫ issus d'un vaste dépôt de documents récemment déclassifiés situé dans les archives numériques sur la sécurité nationale (Digital National Security Archive) [2]. Le premier problème à régler consiste à déterminer le rendement des techniques de pointe de catégorisation de textes proposées lorsqu'on les applique à la classification de sécurité. La recherche doit aussi permettre de déterminer dans quelle mesure ces techniques pourront s'étendre à de vastes collections de données sur différents sujets intéressant différents secteurs de l'organisation. Finalement, grâce à l'examen attentif de la façon dont un système automatisé détermine les données de nature délicate, des progrès pourront être réalisés dans la quête d'une solution au problème connu de l'agrégation de données.

# Table of contents

# Acknowledgements

I wish to thank Dr. Daniel Charlebois for his valuable insights and advice that have helped make this a better paper.

# 1 Introduction

The data-centric security model (DCSM) is a new paradigm for managing the information technology (IT) security of corporate and government data [1]. Rather than traditional models which focus on establishing a secure perimeter for a network that keeps intruders out, a data-centric model ensures that each data item is secured (e.g., through encryption) and access to every item is granted based on a user's identity and level of authorization. Essential to the data-centric model is the requirement that all data in the network be accurately labelled with a security classification (and possibly other metadata) which reflects the appropriate IT security controls surrounding the data. The security classification of a piece of structured or unstructured text [1] is typically assigned by a trained professional who must read the document in question and assess its sensitivity level based on the security policy of the organization. This process can be time-consuming and may result in a lack of consistency depending upon the subjective evaluations of the individuals classifying the documents. If some—or all—of this security classification process were automated, the organization or enterprize could benefit from a more efficient and more *reliable* adherence to security policy.

This paper is intended to present an overview of the problem of automated security classification and to provide a survey of current research in related fields. Automating the process of assigning a classification to a document can be viewed as a problem in text processing and machine learning. Ultimately, the content of a piece of unstructured text must be discerned (e.g., through text processing) and a rule must be applied to assign a security classification to the document in accordance with the security policy (e.g., through machine learning). Text classification or categorization [2] has long been a popular area of research in the field of information retrieval, where data must be assigned to an array of categories to enable easy retrieval based on a topic search [3]. Assigning a security classification, however, is not simply a matter of assigning the document to a particular topic—the security classification may be *related* to the topic, but will not always be determined entirely by the topic. Automated security classification is an area that has not been explored in detail in the literature; however, the related fields of topic identification, document categorization, sentiment classification, and semantic analysis have been extensively studied and will provide a strong basis for initial work in the area of security classification.

The remainder of this paper is organized as follows. Section 2 discusses some of the challenges involved with developing an algorithm for automated security classifica-

1. Note that all data objects require a security classification—not solely text data objects. This report focuses exclusively on text documents with the caveat that other data (e.g., images, audio files) must be assigned a classification as well, although it is beyond the scope of this work.

2. Note that in this paper the terms classification and categorization are used interchangeably when referring to the problem of assigning a piece of text to a specific category.

tion. In Section 3, a survey of related literature is provided with a focus on existing techniques that could be exploited to develop a prototype classification system. Section 4 contains suggestions for future work and further insight into the ultimate utility of research in the field of automated security classification. Finally, Section 5 summarizes the paper and presents conclusions.

# 2 Challenges in Automated Security Classification

While information retrieval and text categorization continue to be active areas of research, there is relatively little in the open literature discussing how these techniques could be applied to security classification (although some notable exceptions are found in [4], [5], [6]). Although similar in many respects to standard text categorization problems, automated security classification presents some unique challenges as well, as discussed below.

Ultimately, the automatic security classification problem can be described as follows: given a set of $m$ documents, $\mathcal{D} = \{d_1, d_2, \ldots, d_m\}$, and a set of $z$ security classifications, $\mathcal{C} = \{c_1, c_2, \ldots, c_z\}$, we want a system that will correctly assign the appropriate classification $c_j \in \mathcal{C}$ to each document $d_i \in \mathcal{D}$. The set, $\mathcal{C}$, of classifications is determined by the organization's security policy. The Government of Canada, for example, has defined four classification levels—unclassified, confidential, secret, and top secret—to designate the level of sensitivity of documents whose contents relate to the national interest [7]. A document that could reasonably be expected to cause injury to the national interest must be classified according to the degree of potential injury: a document that could cause injury is classified as confidential; a document that could cause serious injury is classified as secret; and a document that could cause exceptionally grave injury is classified as top secret. A document which would not be expected to cause injury is unclassified. In addition to classified information, the Government of Canada also specifies that documents whose contents may cause injury to private or non-national interests are designated as Protected, where a level of Protected A, Protected B, or Protected C corresponds to a document where the degree of potential injury is low, medium, or high, respectively.

The standard method for assigning a security classification to a document is for a trained professional to read the document and determine its classification based on the security policy of the organization. One way to automate this process would be to devise a set of rules that could be applied by a computer, which would evaluate every document with respect to the rules and make a decision on its sensitivity (this is sometimes called knowledge engineering). The primary difficulty with this solution is, of course, in determining a workable set of rules. In some cases this might not

be very difficult—an employee's personal information, e.g., social insurance number, should be kept private and this is a simple rule to state. In other cases, however, it might be challenging to define a specific rule that reliably determines whether or not a document might have the capacity to cause injury or might be benign. An alternative to rule-based classification is to use machine learning techniques, whereby a training set of documents with known classifications are processed and the classification rules are learned through inference.

While the security classification structure of an organization may remain fixed (e.g., the possible classifications are unclassified, confidential, secret, and top secret), the security policy which informs exactly which documents should fit into which categories is not static; this presents a challenge for an automated classification system. A changed political climate (e.g., increased vigilance at wartime) may result in a lower threshold being applied for the sensitivity of certain documents; a change in the organizational structure may result in the creation of a new class of documents for which there is no useful historical 'training data' from which to infer a rule set; the passage of time may result in previously sensitive information no longer being deemed sensitive since 'enough water has flowed under the bridge'. Ultimately, security policy is dynamic, and an automated security classification system must be able to adapt to these changes. To some extent, the dynamic nature of security policy may necessitate some human intervention in an automated system to permit the system to adapt. For instance, an automated system might be programmed to automate some of the more obvious classification decisions and to simply *suggest* a classification for documents where the final decision may be less certain—a human arbiter could accept or reject the classification suggestion allowing the system to update its rule set based on inferred policy changes.

The bulk of the research in automated classification and categorization focuses specifically on *topic* classification or categorization—essentially a system attempts to select a topic (or topics) that best describe a document. There are countless applications of this so-called topical classification: an online newspaper service can sort articles based on categories of interest (sports, entertainment, national news); an academic research database may wish to categorize papers based on research topic. Closely related to the work in this field is research in information retrieval, which focuses on matching queries to the most relevant documents by discerning the topic(s) of the query and the topic(s) of the searched sets to find the best match. Assigning a security classification is a fundamentally different problem, since the topic of the document may have some bearing on its sensitivity but is by no means sufficient to determine its ultimate classification level. For instance, a document discussing specific deployment details of Canadian light armoured vehicles may be sensitive and need to be kept confidential; however a newspaper article discussing the use of Canadian light armoured vehicles in Afghanistan is public knowledge and is clearly not classified. Ostensibly, 'Canadian light armoured vehicles' could be identified as the topic of both documents with very

different classifications. Non-topical classification has been studied by Turney [8] and Pang [9], who independently attempted to classify movie reviews as either favourable or unfavourable; Section 3 will discuss this work in more detail.

Another challenge presented by the problem of automatic security classification is in finding appropriate data sets on which to conduct research. Much of the work in information retrieval and text categorization is performed on one or more of several existing corpuses (e.g., the Reuters-21578 corpus, the Reuters Corpus Volume 1 (RCV1), the British National Corpus). To test the validity of a security classification system, however, requires a set of documents containing potentially sensitive information—clearly it is not a simple matter to acquire hundreds or thousands of secret or top secret documents to train a machine learning algorithm.

A final challenge faced when developing and implementing an automated security classification system is non-technical in nature: for the system to be of practical value it needs broad-based user acceptance, meaning that users place some degree of trust in decisions offered by an automated classifier and understand its capabilities and limitations. Although it is unreasonable from a technical perspective to expect an automated system to never make mistakes [3], users may be wary of a classification system which is known to make errors, especially considering the potential consequences of misclassifying documents related to national security. Perhaps if, in addition to an assigned classification, the automated system provided some rudimentary rationale for its decisions (in a form the user could understand) and provided a level of confidence in any particular decision, this might increase system penetration and acceptance. Ultimately, human beings also make mistakes when classifying documents; in fact, an interesting area for future research would be to determine the relative accuracy rates of human classification decisions compared to automated ones. Knowing the relative performances may indeed bolster the case for an automated system.

The dynamic nature of an organization's security policy, the non-topical aspect of security classifications, the difficulty in locating a large corpus on which to conduct experiments, and the psychological aspect of users trusting automated decisions make research into automatic security classification a challenging and interesting problem. The next section includes an introduction to text categorization techniques and a survey of some current research that may provide direction for good first steps.

# 3 Enabling Technologies

The field of natural language processing (NLP) focuses on processing text or speech to discern the underlying meaning and summarize the text, to assign the text to a

3. As discussed in Section 3, some text categorization problems are more challenging than others and success rates of 80 percent might be considered 'good' in these challenging cases.

specific category, to perform information retrieval, or to simplify human-computer interactions. NLP is a broad field incorporating knowledge from linguistics, statistics, psychology, and computer science. This paper focuses specifically on using *statistical* NLP to perform text categorization. In this approach, as opposed to a symbolic NLP approach, individual sentences are not parsed for meaning and the parts of speech do not play a significant role in the analysis. Essentially, each text is treated as a collection of words—the so-called 'bag of words' approach—where word order is not important [4]. This section begins with an introduction to text categorization techniques assuming a bag of words approach and then presents a review of current relevant research in text categorization.

## 3.1 The Mechanics of Text Categorization

In statistical NLP, it is assumed that a corpus (or collection) of documents exists on which a text categorization algorithm can be performed. Typically, these documents have known classifications which can serve as a training set for the text classifier. Documents with known classification are used to train a machine learning algorithm which can then infer a rule to classify new documents from a so-called 'test set'.

Often before advanced processing techniques such as machine learning are applied to each 'bag of words', however, documents must be pre-processed to remove so-called stop words and to perform stemming. Stop words are commonly occurring words that are not expected to lend any value to a classification task, such as 'the', 'an', 'in', 'of', etc. Stemming (or lemmatization) is a process to reduce the occurrence of multiple words that share the same root—for instance a stemming program might reduce the terms 'running', 'runs', and 'ran' all to the root word 'run'.

Other widely used pre-processing techniques are tokenization and grouping. Tokenization refers to the process of further breaking down the terms in documents to potentially useful elements (called tokens) which may have important discriminating power. For instance, the phrase 'does it cost $1?' could be tokenized to the following tokens: 'does', 'it', 'cost', '$', '1', and '?'. As discussed in [11], tokenization may lend more power to the categorization algorithm (depending upon the particular application); in this example, the tokens '$' and '?' indicate that some part of the text may be talking about money and that a question may be posed. If these symbols were buried in a single untokenized term such as '$1?', this meaning may be lost to a categorization algorithm. Grouping, on the other hand, takes similar features and groups them together—for instance, all sets of characters which follow a pattern

4. The bag of words approach is often used because of its ease of implementation. However, Dumais *et. al.* reported in [10] that they saw no improvement in text categorization performance by considering meaningful two-word phrases (e.g., 'New York' as a term versus 'New' and 'York' as separate terms) compared to a simple bag of words.

'xxx-xxxx' (where each 'x' is a number) may be identified as phone numbers. Instead of treating each individual phone number as a separate feature, a grouping algorithm might simply treat 'phone number' as a feature, regardless of the individual digits.

Whether to perform some, all, or any of these pre-processing techniques (i.e., stop word removal, tokenization, stemming, or grouping) depends upon the particular application and categorization problem and is an ongoing area of investigation.

In the standard text categorization literature, a document, $d_j$, is represented as a feature vector, $\mathbf{x_j} = \left(x_j^{(1)}, x_j^{(2)}, \ldots, x_j^{(n)}\right)$, where $x_j^{(i)}$ represents the $i^{\text{th}}$ feature of document $d_j$. In the simplest example, each word or token [5] appearing in the entire corpus of documents may be considered a 'feature'. In this case, if the corpus contained $n$ words, each document $d_j$ would be represented by a vector of length $n$ where each individual term $x_j^{(i)}$ would consist of a term weight $\omega_{ij}$ corresponding to the presence (or relative importance) of word $w_i$ in document $d_j$. A binary weighting scheme, for instance, would have $\omega_{ij} = 0$ if word $w_i$ did not appear in document $d_j$ and would have $\omega_{ij} = 1$ if word $w_i$ did appear in document $d_j$. A popular term weighting scheme is the so-called *tfidf* (term frequency - inverse document frequency) scheme. The weights under *tfidf* are defined as

$$\omega_{ij} = n(w_i, d_j) \cdot \log\left(\frac{|D|}{n(w_i)}\right) \tag{1}$$

where $n(w_i, d_j)$ represents the number of occurrences of word $w_i$ in document $d_j$, $|D|$ represents the number of documents in the corpus, and $n(w_i)$ represents the number of documents in which word $w_i$ occurs. The intuition behind the *tfidf* weighting scheme is that a word appearing many times in a particular document should be weighted more heavily for that document, while a word appearing in many documents in the corpus should be weighted less heavily since the information it provides may not be as useful in uniquely distinguishing document $d_j$.

A typical document corpus might consist of tens or even hundreds of thousands of words. Consequently, the dimensionality of the document vectors $\mathbf{x_j}$ becomes problematic from a computational point of view if all words are retained as features. Thus, after pre-processiong, text classification is usually tackled as a two-step problem:

1. Reduce the dimensionality of the vectors using either feature selection or feature extraction dimensionality reduction techniques (as discussed in Section 3.2);
2. Deliver the reduced-dimensionality vectors to a machine learning algorithm which sorts the documents into categories (as discussed in Section 3.3).

5. For the remainder of the paper, the terms 'word' and 'token' are used interchangeably to refer to features of interest in a feature vector.

## 3.2 Feature Selection and Feature Extraction

To reduce the dimensionality of the document feature set, text categorization techniques typically rely on either feature selection or feature extraction. Feature selection involves defining a utility measure, $A(w, c)$, for each word (or term) $w$ and category $c$, where a document is represent by the $k$ terms ($k < n$) with the largest utility measure. Feature extraction, on the other hand strives to combine existing features in some way to create a new feature space with a reduced dimensionality $k < n$ where the new feature space does not consist of a subset of original features (as in feature selection) but truly consists of new features constructed through manipulation of the original ones.

A careful study of popular *feature selection* techniques was conducted by Yang and Pedersen [12], with the conclusion that the Document Frequency (DF), Chi-Square ($\chi^2$), and Information Gain (IG) techniques (discussed below) were the most effective for text categorization, beating out Mutual Information (MI) and Term Strength (TS) selection. Popular *feature extraction* techniques (discussed below) are Latent Semantic Analysis (LSA)—an algebraic technique that relies on singular value decomposition (SVD)—and Probabilistic Latent Semantic Indexing (PLSI)—a technique similar to LSA that rests on a statistical, rather than algebraic, framework.

### Document Frequency

Document Frequency is arguably the simplest feature selection technique and simply computes the number of documents in the corpus in which a particular term occurs. The terms with the largest frequency counts are kept, while the terms with lower frequency counts are discarded. More formally, a word or term, $w_i$, is retained as a feature if $n(w_i) > \gamma$ where $\gamma$ is some appropriately defined threshold; otherwise it is not retained as a feature of the set. Stop word removal is essential before performing a DF selection to avoid simply being left with a collection of non-informative common words.

Remarkably, DF shows only a minimal loss in performance compared to the more computationally-intensive techniques such as Information Gain and Chi-Square and is a useful dimensionality reduction technique when processing power is at a premium [12].

### Chi-Square

The chi-square ($\chi^2$) technique for feature selection has been found to slightly outperform other feature selection techniques for a number of classifiers when applied to the topical classification of the Reuters RCV1 corpus, as per [13]. Feature selection using $\chi^2$ measures the level of dependence between a category and a word. The idea is to

identify those word-category pairs which are highly dependent, with intuition suggesting that retaining the words from these pairs will lead to greater categorization power.

The $\chi^2$ measure for the word-category pair $(w_i, c_j)$ is computed as

$$\chi^2_{w_i,c_j} = \frac{(n(w_i,c_j)-E(w_i,c_j))^2}{E(w_i,c_j)} + \frac{(n(\overline{w_i},\overline{c_j})-E(\overline{w_i},\overline{c_j}))^2}{E(\overline{w_i},\overline{c_j})} + \frac{(n(\overline{w_i},c_j)-E(\overline{w_i},c_j))^2}{E(\overline{w_i},c_j)} + \frac{(n(w_i,\overline{c_j})-E(w_i,\overline{c_j}))^2}{E(w_i,\overline{c_j})},$$

where $n(w_i, c_j)$ represents the observed number of documents of category $c_j$ in which word $w_i$ appears, $n(\overline{w_i}, c_j)$ represents the observed number of documents of category $c_j$ in which word $w_i$ does not appear, $n(w_i, \overline{c_j})$ represents the observed number of documents which are not of category $c_j$ in which word $w_i$ appears, and $n(\overline{w_i}, \overline{c_j})$ represents the observed number of documents which are not of category $c_j$ in which word $w_i$ does not appear. Similarly, the terms $E(\cdot, \cdot)$ represent the expected number of documents of a particular category in which a word would or would not appear if the word and category were independent. For a word-category pair that is completely independent, the $\chi^2$ metric would evaluate to zero, indicating that the word has no predictive power for the category.

The equation for $\chi^2_{w_i,c_j}$ can be simplified to yield the following:

$$\chi^2_{w_i,c_j} = \frac{|D|\cdot(n(w_i,c_j)n(\overline{w_i},\overline{c_j})-n(\overline{w_i},c_j)n(w_i,\overline{c_j}))^2}{(n(w_i,c_j)+n(\overline{w_i},c_j))\cdot(n(w_i,c_j)+n(w_i,\overline{c_j}))\cdot(n(\overline{w_i},\overline{c_j})+n(\overline{w_i},c_j))\cdot(n(\overline{w_i},\overline{c_j})+n(w_i,\overline{c_j}))} \ . \qquad (2)$$

It is a simple matter to average the $\chi^2$ statistic over all categories to find the marginal $\chi^2$ statistic for each term, where the dimensionality is reduced to $k$ by selecting the $k$ largest statistics.

## Information Gain

The information gain for a word (or token) gives an indication of how many bits of information are gained towards predicting category $c_j$ of a document by knowing that a word appears or does not appear in the document. The information gain for word $w_i$ in category $c_j$ is computed using

$$IG(w_i, c_j) = P(w_i, c_j) \cdot \log\left(\frac{P(w_i, c_j)}{P(c_j) \cdot P(w_i)}\right) + P(\overline{w_i}, c_j) \log\left(\cdot\frac{P(\overline{w_i}, c_j)}{P(c_j) \cdot P(\overline{w_i})}\right), \quad (3)$$

where $P(w_i, c_j) = n(w_i, c_j)/|D|$ denotes the probability that for a random document in $D$ the document belongs to category $c_j$ and contains word $w_i$. Similarly, $P(\overline{w_i}, c_j)$ denotes the probability that a document belongs to $c_j$ and does not contain $w_i$, $P(c_j)$ is the probability that a document belongs to $c_j$ and $P(w_i)$ is the probability

a document contains $w_i$. All probabilities can be computed by counting occurrences in the training set and (3) reduces to

$$IG(w_i, c_j) = \frac{n(w_i, c_j)}{|D|} \cdot \log\left(\frac{|D| \cdot n(w_i, c_j)}{n(c_j) \cdot n(w_i)}\right) + \frac{n(\overline{w_i}, c_j)}{|D|} \cdot \log\left(\frac{|D| \cdot n(\overline{w_i}, c_j)}{n(c_j) \cdot n(\overline{w_i})}\right) \quad . \quad (4)$$

Similar to the $\chi^2$ measure, the IG can be averaged over all categories $c_j \in \mathcal{C}$ to find the marginal IG for a word $w_i$. Choosing the $k$ largest IG values yields the appropriate dimensionality reduction.

## Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a technique introduced by Deerwester et. al. in their landmark paper in 1990 [14]. LSA is a feature *extraction* technique, which represents each document as a $k$-dimensional vector, where individual elements of the vector do not correspond to individual words in the document but correspond to some carefully chosen synthesis or combination of the original words. LSA was introduced as an attempt to overcome the problem of synonymy in document representation and query formulation in information retrieval. For example, a query for documents relating to 'cars' should be able to find a document which uses only the term 'automobile' (even though the document never actually uses the word from the query). The assumption of LSA is that there is some underlying latent semantic meaning to a document that is partially obscured by arbitrary word choice (where other synonyms would have been just as valid). LSA sets out to extract this hidden meaning to find a more useful representation of the document.

To perform a latent semantic analysis on a corpus of documents the first step is to construct a so-called word-document matrix, $X$, where each row of the matrix corresponds to a unique word in the corpus (there are $n$ words in the corpus) and each column of the matrix corresponds to a unique document (there are $m$ documents). If document $d_j$ contains word $w_i$ then entry $X_{ij}$ in the matrix is given an appropriate weighting $\omega_{ij}$ where the choice of $\omega_{ij}$ depends upon the particular term weighting scheme (e.g., binary, *tfidf*). This matrix tends to be *very* sparse since most documents contain only a small subset of the words in the corpus.

LSA next computes the singular value decomposition (SVD) of the $n \times m$ word-document matrix, $X$, where $X$ is decomposed as a product of matrices:

$$X = W_0 S_0 D_0', \quad (5)$$

where $W_0$ and $D_0$ have orthonormal columns and $S_0$ is diagonal and contains the singular values of $X$ (typically $S_0$, $W_0$, and $D_0$ are permuted such that the singular values are ordered from largest to smallest along the diagonal of $S_0$). The $k$ largest

values of $S_0$ are retained and the rest are truncated (i.e., set to zero) to form a new $k \times k$ matrix $S$ such that a new (reduced rank) estimate of $X$ is formed by

$$\widehat{X} = WSD', \tag{6}$$

where $W$ is an $n \times k$ matrix and $D$ is an $m \times k$ matrix, where the rows of $D$ represent the $k$-dimensional representations of the documents. By selecting the $k$ largest singular values, LSA purports to extract the $k$ most representative features of the document.

While LSA has been shown to reduce the problem of synonymy in information retrieval, it has been observed to be less effective as a dimensionality reduction technique for text categorization. This could be because LSA does not take into account the most *discriminative* features of a document (with respect to a particular category), but instead attempts to determine the most *descriptive* features, which may not (in general) be the most discriminative. However, Sebastiani [15] points out that in instances where categorization is determined by incremental contributions of many (possibly hundreds) of terms, then LSA may succeed where feature selection techniques might fail. If, however, a small number of key terms were instrumental in performing a categorization, then LSA would not be the preferred dimensionality reduction technique.

## Probabilistic Latent Semantic Indexing

Probabilistic Latent Semantic Indexing (PLSI), first proposed in [16], uses a statistical approach to reduce the dimensionality of an $n \times m$ word-document matrix ($X$), as opposed to LSA, which uses an algebraic approach. PLSI begins with the assumption that there exist a set of $k$ topics (or more generally $k$ latent variables), $\mathcal{Z} = \{z_1, z_2, \ldots, z_k\}$, where these latent variables are the features that best describe the documents in the corpus. It is assumed that the matrix $X$ represents the occurrence of a particular set of training data. PLSI would like to describe the underlying probability function $P(w_i, d_j)$ which led to this particular observation through the presence of the latent variables.

PLSI models the joint probability of a word-document pair as

$$P(w_i, d_j) = \sum_x P(z_x) \cdot P(w_i|z_x) \cdot P(d_j|z_x), \tag{7}$$

where, conditioned on $z$, it is assumed words are generated independently of a particular document. Furthermore, a set of probabilities, $\theta = \{P(w_i|z_x), P(d_j|z_x), P(z_x)\}$, is defined for notational convenience. The log-likelihood of the observation $X$ given $\theta$ is thus

$$P(X|\theta) = \sum_{i,j} n(w_i, d_j) \cdot P(w_i, d_j). \tag{8}$$

As described in [16], the value $\widehat{\theta} = \arg\max_\theta \log P(X|\theta)$ can be obtained through iteration using the expectation maximization (EM) algorithm. This yields the maximum likelihood value for the set of probabilities, $\theta$, from which $P(w_i, d_j)$ is a direct result. A degree-$k$ representation of a document is thus obtained as a set of probabilities $P(d_j|z_x)$, where $x \in \{1, \ldots, k\}$.

## 3.3 Text Categorization

Text categorization itself is performed after appropriate pre-processing techniques have been applied to the documents and dimensionality reduction through feature selection or feature extraction has reduced the space to a computationally feasible realm. The essence of text categorization is to use a training set of documents with known classifications in order to infer the classification of a new (test) set of documents. Thus, the text categorization problem is essentially one of *supervised* learning, whereby a classifier must learn a rule based on previously correctly categorized documents. This section discusses several popular classifiers from the literature: $k$-Nearest Neighbour, Naive Bayes, Linear Least Squares Fit, and Support Vector Machines.

### k-Nearest Neighbour

The $k$-Nearest Neighbour ($k$-NN) classifier is based upon the intuition that the categorization of training documents which are 'close' to a particular test document (according to some distance metric or measure of closeness) will provide the best categorization for the test document [17]. So, for instance, if the $k$ closest training documents to the test document, $d_j$, are mostly of category $c_i$, the $k$-NN classifier will assign category $c_i$ to $d_j$. A common closeness measure easily applied to documents represented as vectors is the cosine similarity:

$$\cos(d_i, d_j) = \frac{\sum_p \omega_{pi} \cdot \omega_{pj}}{\sqrt{\sum_p \omega_{pi}^2}\sqrt{\sum_p \omega_{pj}^2}}, \tag{9}$$

where $\omega_{pi}$ denotes the weight of word $p$ in document $d_i$.

The computation of the $k$-Nearest Neighbour metric for a new test document $d_j$ and category $c_i$ is given by

$$kNN(d_j, c_i, k) = \sum_{d_z \in \{\text{NN of } \mathrm{d_j}\}_\mathrm{k}} \Delta(d_j, d_z) \cdot \mathbf{I}_{c_i}(d_z), \tag{10}$$

where $\Delta(d_j, d_z)$ is a closeness measure between two documents $d_j$ and $d_z$, and $\mathbf{I}_{c_i}(d_z)$ is an indicator function which is equal to 1 when $d_z$ is classified in category $c_i$ and is equal to 0 when $d_z$ is not classified in category $c_i$. The sum in (10) is taken over

the $k$ documents for which $\Delta(d_j, d_z)$ is largest (essentially the $k$ nearest neighbours of $d_j$). The category $c_i$ for which $kNN(d_j, c_i, k)$ is the largest yields the classification for $d_j$ [6].

From an implementation viewpoint, $k$-NN is arguably the simplest classifier and often forms a baseline for comparison of more advanced classifiers. Of note is that the $k$-NN classifier does not build a classification model in advance based on training data, but simply compares a test document with the known training documents at runtime. This means that the training time is essentially zero, but as the training set increases in size, the classifier will have an increasingly slower performance at runtime.

## Naive Bayes

A Naive Bayes classifier, [3], bases its categorization decisions on the estimated computation of the probability of test document $d_j$ belonging to class $c_i$, specifically, computing $P(c_i|d_j)$. From Bayes' rule,

$$P(c_i|d_j) = \frac{P(c_i) \cdot P(d_j|c_i)}{P(d_j)}. \tag{11}$$

The so-called 'naive' assumption of the NB classifier is that the words in each document occur independently [7]. This assumption allows (11) to be written as

$$P(c_i|d_j) = \frac{P(c_i) \cdot \prod_{1 \leq k \leq |d_j|} P(w_k|c_i)}{P(d_j)}, \tag{12}$$

where $P(w_k|c_i)$ is the probability that, given a document from category $c_i$, it contains word $w_k$. The probabilities in (12) are computed using the relative frequency approach, whereby $P(c_i) = n(c_i)/|D|$ and $P(w_k|c_i) = T_{c_i}(w_k)/\sum_{w \in c_i} T_{c_i}(w)$ (where $T_{c_i}(w_j)$ denotes the number of times word $w_j$ occurs in all documents of class $c_i$, including multiple counts per document). The category which maximizes (12) is chosen as the classification for document $d_j$.

One problem with the formulation of the Naive Bayes classifier as written in (12) is that if document $d_j$ contains any term $w_k$ which has never appeared in $c_i$ in the training set, then $P(w_k|c_i) = 0$ and the product in (12) evaluates to zero regardless of the evidence of the other words in $d_j$. Consequently, Naive Bayes is typically reformulated with a smoothing term such that

$$P(w_k|c_i) = \frac{T_{c_i}(w_k) + 1}{\sum_{w \in c_i} (T_{c_i}(w) + 1)} \ . \tag{13}$$

6. Note that this particular formulation assumes a document may have only one categorization. Alternatively, each category $c_i$ and its converse $\overline{c_i}$ may be applied to the $k$-NN classifier and the largest result chosen in each case.

7. This is clearly untrue—for instance, a document containing the word 'skiing' may be more likely to contain the words 'snow' and 'sport' than the words 'politics' and 'library'.

Naive Bayes performs remarkably well and is very often included in comparisons among state-of-the-art classifiers. It is known to be quite sensitive to feature selection, however, and its performance can degrade rapidly (much moreso than support vector machines) if the features space is reduced significantly [13].

## Linear Least Squares Fit

The Linear Least Squares Fit (LLSF) classifier uses regression to find a model that minimizes the squared error of the classification of the training data. LLSF defines an input matrix, $X$, as the $n \times m$ word-document matrix representing the training set (reminiscent of the matrix $X$ discussed in Section 3.2), a $z \times m$ output matrix, $Y$, representing the correct (binary) classification of the training set (where $m$ documents are sorted into $z$ categories), and a $z \times n$ matrix, $M$.

The classifier computes the matrix $\widehat{M}$ such that

$$\widehat{M} = \arg\max_{M} \|MX - Y\|^2. \tag{14}$$

Faced with a new test document, $d_j$, the LLSF computes $Y_{d_j} = \widehat{M}X_{d_j}$, where $X_{d_j}$ denotes the $n \times 1$ vector representing document $d_j$ and $Y_{d_j}$ represents the $z \times 1$ vector of $z$ categories, providing an estimate of the membership of $d_j$ in each category.

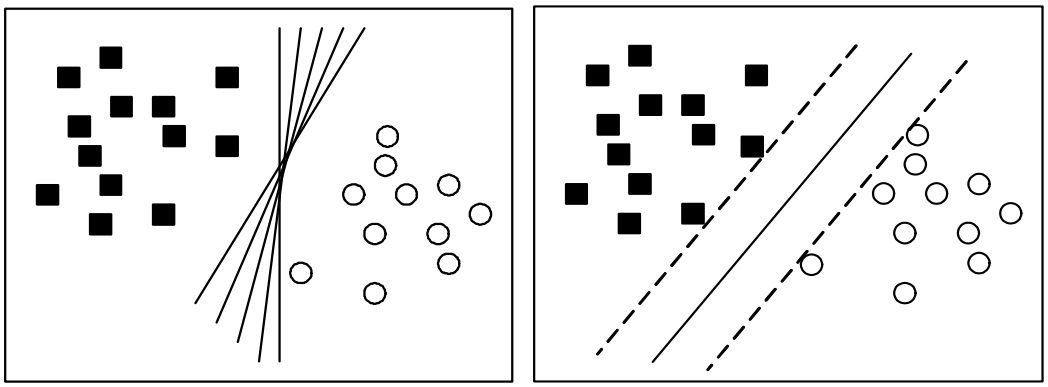
The LLSF has been found to be comparable in performance to some of the best classifiers implemented using support vector machines, as discussed in [18].

## Support Vector Machines

Support Vector Machines (SVM) have been identified as a powerful tool for text categorization, first exploited by Joachims in [19]. The most common application of an SVM to text categorization is in making a binary decision about the categorization of a document (e.g., the document is either a member of category $c_1$ or it is a member of category $c_2$; alternatively, the document is a member of category $c_i$ or it is not). Assuming that the training data is linearly separable [8], an SVM finds the hyperplane separating the two groups that is furthest away from any training vector. Fig. 1 depicts an example where the training data vectors are represented by points in a 2-dimensional plane. In Fig 1(a) it is clear that there are many hyperplanes (in this case the hyperplane is simply a line) which separate the two data sets. Many of these separating lines, however, come very close to some of the data vectors whereby

8. There are methods to adjust the SVM optimization when the data is not linearly separable. In the case where some data falls on the wrong side of the decision boundary, a penalty function can be assigned. In the case where the boundary is non-linear, the so-called Kernal Trick can be applied. These methods are discussed further in [20].

some small noise in the data might result in a misclassification if the point lay on the opposite side of the separation line. The hyperplane constructed by the SVM is intended to provide the widest margin of any of the possible hyperplanes as shown in Fig. 1(b), where the solid line is the SVM hyperplane and the two dotted lines delineate the margins on either side known as the 'supporting' planes.



(a) (b)

***Figure 1:*** *SVM finds the hyperplane with the widest margin*

Following the standard notation of [20], assume that the training set consists of $L$ training points (i.e., vectors), $\mathbf{x}_i$, of dimensionality $n$ (here, the $\mathbf{x_i}$ terms can be thought of as the dimensionality-reduced document vectors) where each point is labelled with a classification $y_i$ (where $y_i \in \{+1, -1\}$ denotes a binary classification). Assume further that the data is linearly separable by an $(n-1)$-dimensional hyperplane. A hyperplane can be defined by $\mathbf{w} \cdot \mathbf{x} + b = 0$, where $\mathbf{w}$ is a vector normal to the hyperplane and $b/\|\mathbf{w}\|$ defines the distance of the hyperplane from the origin. The SVM classifier applies the decision function $f(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$ to determine the classification of any test vector $\mathbf{x}$. The output of the decision function is invariant to a positive rescaling of the arguments of the sign function (i.e., a rescaling of $\mathbf{w}$ and $b$), so it is possible to implicitly fix the scaling such that

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b &\geq +1 \quad \rightarrow \quad y_i = +1 \\ \mathbf{x}_i \cdot \mathbf{w} + b &\leq -1 \quad \rightarrow \quad y_i = -1. \end{aligned} \tag{15}$$

With the decision function rescaled as in (15), the two supporting hyperplanes are separated by a distance or *margin* of $2/\|w\|$. The central problem of the SVM, then, is to find $\mathbf{w}$ and $b$ such that the margin $2/\|w\|$ is maximized and the constraints in (15) are observed. This can be succinctly written as

$$\begin{aligned} \min_{w,b} \quad & \tfrac{1}{2}\|w\|^2 \\ s.t. \quad & y_i \cdot (\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1. \end{aligned} \tag{16}$$

The optimization problem in (16) can be solved using Quadratic Programming and Lagrange multipliers with details in [20] and [21].

Since its first application in [19], SVM has become the benchmark against which other text categorization methods are measured—to date its performance has been difficult to exceed.

## 3.4 The State of the Art

Sections 3.2 and 3.3 provided an overview of popular dimensionality reduction techniques and text categorization algorithms. Although many of the techniques presented have been known in the machine learning community for years, their study continues to be an active area of research in statistical natural language processing and these techniques are in use in today's existing commercial text categorization applications. In fact, for topic identification (in particular binary topic identification, i.e., choosing between two possible topic categories), support vector machines represent the current state of the art. Much of the current research focuses on refining known techniques to improve accuracy or to reduce processor requirements; also of interest to the research community is the application of known techniques to new problems such as non-topical classification, which will be discussed in Section 3.5.

Since Yang first compared feature selection techniques in a systematic way in [12] and [13], a number of papers have repeated the study with certain variations (e.g., change the learner, propose new untested feature reduction schemes) while ultimately coming to a similar conclusion. In [22], a different approach was taken by studying feature selection at both a global level and a local level. Global level feature selection is the technique described in Section 3.2 and studied by Yang whereby the $k$ most discriminating features of the entire corpus are selected. In local feature selection, however, the $k$ feature that most discriminate for each particular category are computed. Clearly, for a corpus with many categories, each category may well have a different set of $k$ features which are most discriminatory for that particular category. Thus, using local feature selection, documents are no longer represented as a vector of features, but are represented as a set of vectors of features where each vector corresponds to a particular category. Ultimately, [22] finds that when only a small number of features are retained a local feature selection policy based on the chi-square technique results in better performance; when a large set of features are retained, however, the standard global feature selection policy with chi-square feature selection is best.

Recently, [23] reported the results of a new study that conducted an extensive review of *term weighting* schemes. This study focuses not on which feature selection method is preferable, but instead attempts to determine how individual features should be weighted once those features have been selected (using one of the methods from

Section 3.2). They evaluate the schemes by determining the text categorization performance of an SVM on the Reuters-21578 corpus using ten possible term weighting schemes including simple binary weighting (i.e., all features have the same weight) and the popular *tfidf* weighting. The study concludes that the choice of term weighting depends upon the number of features retained (where the chi-square technique is used for feature selection). For a small number of retained features it is found that a $tf \cdot chi$ scheme [9] performs far worse than any other term weighting scheme and the other nine schemes considered are comparable. With a high number of retained features, the poorest performing term weighting schemes are binary and *tfidf*. It is noted, however, that the particular corpus used will play a large role as well, although this is not the focus of the paper.

Current work on $k$-NN classifiers for text categorization is focused primarily on methods to improve the classification speed. The standard $k$-NN implementation has no 'training phase'—new test documents are simply compared to the stored training documents at run time, leading to potentially long processing times. In [24], a solution called Semantic-Centre $k$-NN is proposed whereby the training set is processed prior to run time to build a so-called semantic space. Essentially, clusters of closely-spaced documents (i.e., documents in the test set with a large cosine similarity measure) are collapsed into a single point represented by the centroid of the documents. New documents requiring categorization are then compared to the collapsed points instead of the actual training documents. Ultimately, this results in a trade-off between the categorization accuracy and the degree of compression of the training set (since clearly if the training set is collapsed by too great an extent, then the neighbors may not be sufficiently representative of the categories). It is reported in [24] that a reduction in runtime by a factor of ten results in only a one percent degradation in precision and recall performance compared to standard $k$-NN. While the method in [24] trades performance for speed, [25] proposes an algorithm that sees no degradation in $k$-NN performance but provides a fixed (and thus a non-configurable) increase in speed. In [25], the training set is pre-processed by creating a tree which allows for the search of the $k$ nearest neighbours of any given test document. In this fashion the $k$ nearest neighbours of a document are found at a speed comparable to searching through a tree as opposed to searching through every training document one by one, resulting in dramatic increase in speed, especially for small $k$.

Much of the current research on support vector machines is focused on reducing memory requirements and processing time, and on improving the efficiency of multiclass SVMs. The standard SVM formulation creates a hyperplane that divides a set of training data into *two* halves, resulting in an inherently binary classifier. Since many classification problems involve multiple categories ($N > 2$), methods to use SVMs

9. The $tf \cdot chi$ scheme multiplies the chi-square metric by the relative frequency of the term in the document.

in these cases are of interest and a nice summary is provided in [26]. Popular multi-class methods are one-vs-all and one-vs-one. For the one-vs-all method, $N$ SVMs are constructed where the training examples in category $i$ are used as positive instances and all other training examples are used as negative instances when constructing the $i^{\text{th}}$ SVM. A particular test document is tested against each of the $N$ SVMs and the document's class is determined by the SVM that gives the highest predictive value. For the one-vs-one method, every distinct pair of categories is used to train a separate SVM (so $N(N-1)/2$ SVMs are trained in total). A test document is tested against each SVM and the class with the highest number of positive votes is identified as the predicted class for the document. A multitude of research papers continue to propose methods to improve multi-class SVM efficiency, often sacrificing a small degree of precision/recall performance for large processing and memory gains—see [27] and [28] for typical examples.

The well known and well established techniques for dimensionality reduction and machine learning discussed in Sections 3.2 and 3.3 form a solid foundation for text categorization through statistical means with a significant proportion of current research focusing on improving the training time, accuracy, and runtime performance of these algorithms. Another branch of research focuses on applying these techniques to problems in non-topical text categorization, which is the subject of the next section.

## 3.5 Non-Topical Text Categorization

The majority of the research in text categorization has focused on *topical* categorization, where a classifier attempts to assign a document to a category based on subject matter, typically using the techniques discussed in Sections 3.2 and 3.3 or some variation on these. As discussed in Section 2, determining a security classification is not strictly limited to identifying the topic of a passage of text. This section discusses some of the recent research in the field of non-topical text classification.

### Unsupervised Classification

In [8], Turney introduced the problem of sentiment classification—essentially, the intent was to find a method to determine if a customer review of a particular product or service was positive or negative; in other words, classify a customer review as 'recommended' or 'not recommended'. A corpus of customer reviews from four domains—movies, automobiles, banks, and travel destinations—was used for the study. The approach taken to classify the reviews was to identify key words or phrases which might be likely to influence the sentiment of the review; Turney selected bigrams (i.e., two-word phrases) containing adverbs and adjectives and assigned a sentiment score to each of these bigrams. Documents with a positive sentiment score were deemed to

'recommend' the product under review and documents with a negative score to 'not recommend' the product.

To determine the sentiment score of a particular bigram, [8] uses a method called Pairwise Mutual Information through Information Retrieval (PMI-IR). The basis of this method is to determine the mutual information between a particular bigram and the term 'excellent' and between the bigram and the term 'poor'. The mutual information between a bigram and the term 'excellent' is computed by performing Internet searches (hence the moniker 'information retrieval' in PMI-IR) using a standard search engine to evaluate

$$\text{PMIIR}(\text{bigram}, \text{`excellent}') = \frac{\text{h}(\text{bigram AND `excellent}')}{\text{h}(\text{bigram})\text{h}(\text{`excellent}')}, \tag{17}$$

where $h(x)$ represents the number of hits generated by a search for the term $x$. The PMI-IR for 'poor' is computed similarly. The ratio of the PMI-IR scores determines the sentiment score of the bigram.

Turney's method showed a classification accuracy as high as 84 percent for automobile reviews but only 65 percent for movie reviews. Of note is that the classification method used is unsupervised, since the decision is based solely on PMI-IR scores, rather than on decision methods by a trained learner as discussed in Section 3.3.

## Supervised Classification

Pang *et. al.* [9] studied the classification of customer reviews specifically for movies (the domain that fared worst in Turney's scheme) using supervised learning methods. They evaluated Naive Bayes and SVM text categorization methods on a corpus of movie reviews, where once again the classification problem was binary: 'recommended' or 'not recommended'.

Whereas Turney extracted two-word sentiment phrases from the reviews, [9] initially performed no feature selection or feature extraction and kept all the words (i.e., all the unigrams) in each document without reducing the dimensionality. The only pre-processing performed was to obtain a frequency count for words in the documents. Direct application of Naive Bayes and SVM learners resulted in a categorization accuracy of 79 percent and 73 percent respectively, with Naive Bayes actually outperforming SVM and both methods outperforming the results in [8].

In an attempt to improve the performance of their classifier, Pang *et. al.* experimented with a number of pre-processing techniques. First, they eliminated the frequency count and simply focused on whether or not a word was present in a document. This turned out to be the most advantageous scenario with Naive Bayes and SVM reaching accuracies of 81 percent and 83 percent respectively. They argued that by

heavily weighting the frequency of words, the classifier became distracted by the topic of the text rather than the sentiment (where they hypothesized that there are fewer sentiment words in a text than topic words). They also studied the effect of including bigrams in addition to unigrams, focusing only on adjectives, and of taking into account where in the document a word was used (beginning, middle, or end). None of these changes resulted in any improvement over the simple unigram implementation.

## Supervised Classification Across Domains

One conclusion of [8] was that a single classification algorithm is not equally effective across a number of different domains (e.g., movie reviews versus automobile reviews). Further research in [29] clearly demonstrated the benefit of intra-domain training examples in order to successfully classify a new test document in a particular domain. For instance, the best way to classify the sentiment of a movie review is to have a large training corpus of movie reviews as opposed to a large corpus of reviews in some other domain. The importance of intra-domain data is explained by the observation that key words describing sentiment may vary in interpretation across domains; if a movie is described as 'unpredictable' this might be considered a positive trait, while an 'unpredictable' automobile is certainly not favourable.

While the ideal multi-domain classification solution is to have a rich training set with many examples in all domains of interests this is not always feasible. In [29], several approaches to classifying data in a 'new' domain (a domain with very little training data) were proposed and analyzed. One favourable approach was the so-called ensemble approach. In the ensemble approach the new domain was classified by using a weighted combination of the output of classifiers from known domains, where the weights were chosen to maximize the performance of the ensemble in the target domain [10]. This resulted in an improvement over a single all-purpose classifier (the so-called 'all-data' approach) but no method was able to outperform individual classifiers specifically tuned to a particular domain based on a rich set of training examples.

Another approach to the cross-domain classification problem was presented in [30]. In [30], the authors computed the *sentiment score* of all words in the cross-domain training set in order to identify words that most strongly predicted either a positive or negative review. A test document from a new domain was analyzed to determine the three keywords it contained with the largest sentiment score. These three keywords were treated as a query and an off-the-shelf search engine was used to search the training set for the documents most related to this query. The classification of these training documents was used to classify the new test document. This method also

10. It is assumed that a small amount of labelled data in the target domain is available to tune the weights of the ensemble classifier.

outperforms the all-data approach, although comparison to the ensemble approach of [29] is complicated by a different choice of corpus. Ultimately the selection of the three words with the largest sentiment score can be viewed as an extremely aggressive form of feature selection.

## Spam Filtering

A familiar text classification problem is the issue of filtering unwanted spam email. There is a non-topical element to the spam classification problem as the topic of an email may not be the only factor in identifying an email as 'spam' or 'not spam'. The presence of certain phrases or punctuation unrelated to the topic (e.g., 'FREE MONEY', '!!!!!') are often enough to identify a particular email as spam [31]. In fact, spam is often readily identified by noting the presence of key words that have a high likelihood of appearing in spam and a low likelihood of appearing in legitimate email.

Several non-machine learning techniques have been applied with some success in spam filters, such as blocking email from known spam sources (so-called 'blacklisting'), analyzing email delivery patterns (number of recipients, temporal locality among recipients), and creating rule-based filters through heuristics. However, Bayesian machine learning techniques now form a part of nearly all spam filters and are a *de facto* state-of-the-art technology [32]. The Bayesian systems maintain a table of document features (words or tokens) and the associated probability that the feature is indicative of a spam email or a non-spam email—essentially $p(w|spam)$ and $p(w|\overline{spam})$. Most spam filters combine these conditional probabilities using the assumption that they are independent, essentially implementing a Naive Bayes classifier (see Section 3.3 for a discussion of the Naive Bayes classifier). The Bayes statistics are simple to update based on new labelled data provided by user feedback when a user classifies an email as spam or non-spam, resulting in a filter that can tailor its performance to individual users.

Rios and Zha evaluated the performance of support vector machines for spam detection in [33]. They found that the SVM outperformed a Naive Bayes classifier in correctly detecting spam (i.e., the SVM had a higher true positive rate) for a given false positive rate (i.e., for a given number of valid emails misclassified as spam). Of note is that the Naive Bayes classifier performed extremely poorly for small false positive rates but its performance improved, approaching (but not exceeding) the SVM performance as the allowable false positive rate was increased. For spam detection in particular, a small false positive rate is vital since users do not want to lose valid email that is misclassified as spam.

# 4 Next Steps

Section 3 discussed current techniques used for the topical and non-topical classification of text. The application of these techniques to the problem of security classification is largely untried and presents an excellent opportunity for further research. As noted in Section 2, one of the challenges of conducting research in automated security classification is in obtaining a usable document corpus. One potential solution to this problem is the recent online availability of thousands of declassified U.S. Government documents at the Digital National Security Archive [2]. This archive contains documents formerly classified as Secret, Top Secret, and Unclassified across a multitude of topic domains. These documents provide an excellent training and test corpus for initial research in this field.

The first open problem in the field of automated security classification is in identifying which of the current known text categorization techniques is best suited to the problem of security classification. Beginning with documents taken from a single topic domain, some research avenues of interest are detailed below.

- **Dimensionality Reduction**: Dimensionality reduction techniques (see Section 3.2) have been examined for topical text categorization and their performance compared on a number of corpora. As noted in [9], some of the commonly held notions about term weights (e.g., *tfidf*) and frequency selection do not apply in non-topical sentiment classification. Of interest is which dimensionality reduction techniques are best suited to the task of automated security categorization. Additionally, while feature selection techniques tend to predominate the literature of text categorization, feature extraction techniques may well prove advantageous for security classification as the decision may rely on the incremental contributions of many terms (as opposed to a small number of influential terms).
- **Categorization Method**: The choice of categorization method (as discussed in Section 3.3) is a topic on which extensive research has been conducted for topical text categorization. An important research avenue is to apply the same rigor to automated security classification in order to determine which categorization algorithm is best suited to the classified corpus and to identify which dimensionality reduction techniques are best suited to which categorization algorithm.
- **Non-topical Heuristics**: Turney used a heuristic technique to identify phrases that might help classify the sentiment of a document in [8] (to mixed success), and spam filters commonly use a pre-packaged set of key words to identify unwanted email (in addition to using machine learning) [31]. This suggests that some rule-based knowledge engineering might be useful as a supplement to the machine learning text categorization algorithms studied. Careful analysis of the labelled document corpus may suggest some *simple* rules which could improve the performance of automated security classification—comparing this assisted performance

to the unassisted performance is another area of interest.

While it is sensible for initial research into automated security classification to focus on identifying the best performing dimensionality reduction and text categorization techniques for a single topic domain, additional research should focus on important questions that need to be addressed in order to successfully deploy a functional classification system. Some important questions and avenues for further research are as follows.

- **Incorporating New Intra-Domain Learning**: A useful incarnation of an automated security classification system could be a system which *suggests* a classification for a new document and allows the user to provide the final decision. How easily does the system allow for the incorporation of new (intra-domain) information and how heavily does it weight the information gained when a user disagrees with a suggested classification? Related to this question is the ability of a system to adapt to changes over time. By what factor would the accuracy of a system trained with documents from the 1970s degrade when the test set consisted of documents from the 1980s? How quickly could the system adapt?
- **Cross-Domain Classification**: As shown in [29] and [30], non-topical sentiment classification across domains proves problematic. With security classification, it is reasonable to expect that this might also be the case. For instance, the distinguishing features which determine the security classification of a document might be vastly different when a document concerns U.S. foreign policy towards the Philippines as opposed to U.S. nuclear disarmament strategy. How effectively would an 'all-data' system perform when classifying documents across multiple domains? Are there effective strategies to apply known domain data to a new domain? How quickly will the system be able to incorporate a new domain through online (user-feedback based) learning?
- **Data Aggregation**: A known problem faced by organizations with a hierarchy of security classifications is the data aggregation phenomenon. Essentially, certain documents may not be considered 'sensitive' when viewed individually and so may be unclassified; however, when collected together the aggregate of these individual documents may convey (possibly through inference) information of a sensitive nature and the aggregate *should* be classified. Identifying which aggregates constitute information of a higher security classification is far from a simple matter. Conducting research in automated security classification may yield a model for the data which could be exploited to give some understanding of how data are combined to produce a document more sensitive than the sum of its parts. A simple application of the automated classification system to the problem of data aggregation might be to apply the system to various subsets of a collection of documents to identify which subsets yield a document of higher sensitivity (according to the automated classifier). Even this simple application would be of value if it led to a method for identifying sensitive aggregates.

While the research ideas proposed in this section are far from exhaustive, it is clear that automated security classification is a field with many open questions and interesting avenues for investigation.

# 5 Conclusion

This paper presented an overview of important technical considerations involved in developing a system for automatically assigning a security classification to a document. Some of the challenges of automating the process of security classification were discussed—specifically, the dynamic nature of an organization's security policy, the non-topical nature of security classification, and the difficulty in obtaining a large corpus on which to conduct experiments. A survey of current state-of-the-art text categorization technology was provided which discussed popular feature selection and feature extraction techniques for dimensionality reduction as well as effective machine learning algorithms for topical classification. Non-topical text classification was discussed with a focus on sentiment classification and spam filtering.

Open avenues for future research in the area of automated security classification were presented with the conclusion that there is still much to learn in this field. While developing an automated system is an interesting problem in its own right, research in this field may lead to new discoveries addressing problems in the field of data aggregation and security leaks through inference channels.

# References


[1] Grandison, T., Bilger, M., O'Connor, L., Graf, M., Swimmer, M., Schunter, M., Wespi, A., and Zunic, N. (2007), Elevating the Discussion on Security Management: The Data Centric Paradigm, In *2nd IEEE/IFIP International Workshop on Business-Driven IT Management*.

[2] Digital National Security Archive (online), `http://nsarchive.chadwyck.com/home.do` (Access Date: September 8, 2009).

[3] Manning, C., Raghavan, P., and Schutze, H. (2009), An Introduction to Information Retrieval, Cambridge University Press.

[4] Blakley, Bob, Information Classification: The Most Important Security Thing You're (Still) Not Doing (online), `http://www.burtongroup.com/Client/Research/ServiceHome.aspx?service=SRMS` (Access Date: August 27, 2009).

[5] Clark, K. (2008), Automated Security Classification, Master's thesis, Vrije Universiteit.

[6] Mathkour, H., Touir, A., and Al-Sanie, W. (2004), Automatic Information Classifier Using Rhetorical Structure Theory, In *IIWAS*, Jakarta.

[7] Government Security Policy (online), `http://www.tbs-sct.gc.ca/pol/doc-eng.aspx?id=12322&section=text` (Access Date: August 26, 2009).

[8] Turney, P. (2002), Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews, In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia.

[9] Pang, B., Lee, L., and Vaithyanathan, S. (2002), Thumbs Up? Sentiment Classification Using Machine Learning Techniques, In *Proceedings of EMNLP*.

[10] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998), Inductive Learning Algorithms and Representations for Text Categorization, In *In Proc. of CIKM-98, 7th ACM Internation Conference on Information and Knowledge Management*, pp. 148–155, Washington.

[11] Khoo, A., Marom, Y., and Albrecht, D. (2006), Experiments with Sentence Classification, In *Proceedings of the 2006 Australasian Language Technology Workshop*.

[12] Yang, Y. and Pedersen, J. (1997), A Comparative Study on Feature Selection in Text Categorization, In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, Nashville.

[13] Rogati, M. and Yang, Y. (2002), High-Performing Feature Selection for Text Classification, In *Proceedings of the 11th international conference on Information and Knowledge Management*.

[14] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990), Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science*, 41(6), 391–407.

[15] Sebastiani, Fabrizio (2002), Machine learning in automated text categorization, *ACM Computing Surveys*, 34(1), 1–47.

[16] Hofmann, Thomas (1999), Probabilistic Latent Semantic Analysis, In *In Proc. of Uncertainty in Artificial Intelligence, UAI99*, pp. 289–296.

[17] Yang, Y., Expert Network: effective and efficient learning from human decisions in text categorization and retrieval, In *In Proc. of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, Dublin.

[18] Yang, Y. and Liu, X. (1999), A Re-Examination of Text Categorization Methods, In *In Proc. of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pp. 42–49.

[19] Joachims, T. (1998), Text Categorization with Support Vector Machines: learning with many relevant features, In *In Proc. of ECML-98, 10th European Conference on Machine Learning*, pp. 137–142, Heidelberg.

[20] Bennett, K. and Campbell, C. (2000), Support Vector Machines: Hype or Hallelujah?, *SIGKDD Explorations*, 2, 1–13.

[21] Fletcher, T., Support Vector Machines Explained (online), `http://www.cs.ucl.ac.uk/staff/T.Fletcher` (Access Date: September 3, 2009).

[22] Tasci, S. and Gungor, T. (2008), An Evaluation of Existing and New Feature Selection Metrics in Text Categorization, In *In Proc. of 23rd International Symposium on Computer and Information Sciences*, Istanbul, Turkey.

[23] Lan, M., Sung, S., Low, H., and Tan, C. (2005), A Comparative Study on Term Weighting Schemes for Text Categorization, In *In Proc. of International Joint Conference on Neural Networks*, Montreal, Canada.

[24] Zhang, X., Huang, H., and Zhang, K. (2009), KNN Text Categorization Algorithm Based on Semantic Centre, In *In Proc. of International Conference on Information Technology and Computer Science*, Beijing, China.

[25] Wang, Y. and Wang, Z. (2007), A Fast KNN Algorithm for Text Categorization, In *In Proc. of 6th International Conference on Machine Learning and Cybernetics*, Hong Kong.

[26] Hsu, C. and Lin, C. (2002), A Comparison of Methods for Multiclass Support Vector Machines, *IEEE Trans. on Neural Networks*, 13, 415–425.

[27] Shao, F., He, G., and Zhang, X. (2008), An Improved Algorithm for Multiclass Text Categorization with Support Vector Machines, In *In Proc. of International Symposium on Computational Intelligence and Design*, Wuhan, China.

[28] Dong, J., Suen, C., and Krzyzak, A. (2008), Effective Shrinkage of Large Multi-Class Linear SVM Models for Text Categorization, In *In Proc. of International Conference on Pattern Recognition*, Tampa, Florida.

[29] Aue, A. and Gamon, M. (2005), Customizing Sentiment Classifiers to New Domains: A Case Study, In *In Proc. RANLP, Recent Advances in Natural Language Processing*, Bulgaria.

[30] Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998), A Bayesian Approach to Filtering Junk E-Mail, In *In Proc. of the 15th National Conference on Artificial Intelligence*, pp. 55–62, Madison.

[31] Aue, A. and Gamon, M. (2005), Customizing Sentiment Classifiers to New Domains: A Case Study, In *In Proc. RANLP, Recent Advances in Natural Language Processing*, Bulgaria.

[32] Hunt, R. and Carpinter, J. (2006), Current and New Developments in Spam Filtering, In *In Proc. of 14th IEEE International Conference on Networks (ICON2006)*, Singapore.

[33] Rios, G. and Zha, H. (2004), Exploring Support Vector Machines and Random Forests for Spam Detection, In *In Conf. on Email and Anti-Spam*.

## DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

   Defence R&D Canada – Ottawa
   3701 Carling Avenue, Ottawa, Ontario, Canada
   K1A 0Z4

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

   UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

   Developing an automatic document classification system

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

   Brown, J.D.

| 5. DATE OF PUBLICATION (Month and year of publication of document.) | 6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.) | 6b. NO. OF REFS (Total cited in document.) |
|---|---|---|
| January 2010 | 38 | 33 |

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

   Technical Memorandum

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

   Defence R&D Canada – Ottawa
   3701 Carling Avenue, Ottawa, Ontario, Canada K1A 0Z4

| 9a. PROJECT NO. (The applicable research and development project number under which the document was written. Please specify whether project or grant.) | 9b. GRANT OR CONTRACT NO. (If appropriate, the applicable number under which the document was written.) |
|---|---|
| 15bc05 | |

| 10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) | 10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.) |
|---|---|
| DRDC Ottawa TM 2009-269 | |

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

   (X) Unlimited distribution
   ( ) Defence departments and defence contractors; further distribution only as approved
   ( ) Defence departments and Canadian defence contractors; further distribution only as approved
   ( ) Government departments and agencies; further distribution only as approved
   ( ) Defence departments; further distribution only as approved
   ( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Assigning a security classification to a document is typically a labour-intensive manual process performed by a trained professional who must read and understand the document and subsequently apply the rules of an organization's security policy. Automating this process would increase organizational efficiency and would nicely complement newly proposed data-centric security systems where all data must be accurately labelled with the appropriate classification. This paper introduces some of the challenges faced in developing an automated security classification system and discusses current text categorization technologies (dimensionality reduction and machine learning techniques) which would be the key enablers of such a system. In addition to the technology review, several avenues of research are proposed to evaluate a number of potential solutions to the security classification problem.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. *e.g.* Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

security classification
text categorization
trusted labelling

**Defence R&D Canada**

Canada's leader in Defence
and National Security
Science and Technology

**R & D pour la défense Canada**

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale


**www.drdc-rddc.gc.ca**