An Introduction to Unsupervised Document Classification

Patrick Trinkle
Dept. of Computer Science and Electrical Engineering,
University of Maryland Baltimore County,
Baltimore, MD, 21250
tri1@umbc.edu

May 12th, 2009

Abstract

Unsupervised document classification addresses the problem of assigning categories to documents based solely on their content without a training set and predefined categories. There are many approaches to the problem, which we discuss here. Research is very active in the field of data classification because solutions to this set of problems can provide significant improvement for information retrieval, document filtering (semi-supervised), corpus navigation, and effective news routing. We introduce the vector space model for representing documents, as well as different algorithms for choosing a feature set or term space. We also introduce classifiers including Naïve Bayes, K-Nearest Neighbor, Centroids (Rocchio), Support Vector Machines, N-grams, and briefly Neural Networks.

1 Motivation

Unsupervised document classification addresses the problem of assigning categories to documents based solely on their content without a training set and predefined categories. Effectively this places each document into a specific category or set of categories also known as classes. In semi-supervised and supervised classification there is a training set and a known set of classes. Many approaches in these neighboring fields apply directly to unsupervised classification. Text categorization is commonly utilized to: determine if an email is spam (unsolicited) or the alternative; speedup query processing; provide additional relevant results for a query; allow for a rich method of corpus navigation; and route or filter news by topic.

Clustered documents are understood to be related by content and therefore likely relevant to each other. This notion provides that an information retrieval system can provide better results to a user by returning all the documents in a cluster. Also an information retrieval system can return results faster if the corpus has well-seperated cateogries.

Spam detection is a supervised problem of text filtering whereby an input document is compared against a training set. This form of text filtering merits its mention because of its success in classifying a document in a binary fashion as belonging to a class or not, which is a problem applicable to unsupervised document classification.

Automatically sorting documents by content can also provide topical news information to users based on their preferences. Because news articles are often very specific to a particular topic, they lend themselves to the notion of grouping/classifying.

2 Document Classification Methods

To classify a document as belonging to one class or category you must represent the document in a meaningful way to the classifier. There are a few well principaled approaches to this problem. Many involve representing a document as a vector space of terms in multi-dimensional space, or the bag of words model.

Once the documents are represented appopriately, various algorithms are applied that can group the documents together into classes. Or simply determine if a document belongs in a class or not (a binary approach). In spam filtering this typically involves

scanning a document and comparing its contents to a training set of positive (spam) and negative examples to determing if it is spam. There are also training sets with only positive examples representing a category.

Document clustering can provide better search results as well as a possible speedup in result retrieval. The speedup leverages on the detail that the documents are already in clusters according to their content; therefore a query matching one document in the cluster is likely relevant to the others. This is also how better search results can be provided. The value of the results of a query are measured in two fundamental factors: precision and recall. Precision is defined as the ratio of the documents returned that are relevant to the query veruse those that are not. Recall is defined as the ratio of how many relevant documents were returned versus how many relevant documents are in the corpus [Mann2008].

Regardless of method chosen to cluster the documents or assign classes a classifier is used. The job of the classifier is simply to assign a category to an input document, typically based on predefined features. The choice of features is important and which classifer is used can steer the desired feature set. Some classifiers can only work with small feature sets; while others have the opposite requirement.

3 Document Representation

Documents are typically represented in vector space as multi-deminsional vectors such that each dimension represents a term or feature. This is the bag of words model, where the order of the words in a document is disregarded. The documents are vectors, however the meaning of the distance between the vectors is misleading. To reduce the vector space or term space size stop words are not included in a document vector and only the top m terms or keywords are used. Choosing the top m terms is a problem for feature selection.

For n documents the following matrix represents the vector space model where tw is the term weight for a given term i in document j [Mann2008]:

$$\begin{bmatrix} tw_{i,j} & \cdots & tw_{i,n} \\ \vdots & \ddots & \vdots \\ tw_{m,j} & \cdots & tw_{m,n} \end{bmatrix}$$

3.1 Tf-idf

The Term Frequency Inverse Document Frequency weighting is a method that can be used to build a vector representation of a document. The term frequency used in Tf-idf is the normalized raw term frequency within the document. There are a variety of proven normalization methods, including but not limited to the square root of the sum of squares of term frequencies, or dividing by the length of the document. This normalization is done to balance out the term frequencies in the event of a large document with few occurrences of the term versus a small document with many occurences. Often the Tf-idf representation of the document as a vector is actually the logarithm of the values for the term frequency and the inverse document frequency. This reduction is used to dampen spikes in the data. Given a term i in a document in a corpus of size N the term weight, tw, is defined as follows [Mann2008]:

$$\begin{aligned} tw_i &= tf_i * idf_i \\ tf_i &= \log(termfrequency_i + 1) \\ idf_i &= \log\left(\frac{N}{documentfrequency_i + 1}\right) \end{aligned}$$

The addition of 1 to the value provides that you will never divide by 0 or attempt to calculate the logarithm of 0.

3.2 Binary

The binary representation of a document is considerably simpler than the Tf-idf variant. Given some the top m keywords, the binary representation of a document is sufficiently described with the following, given t_i is is the term [Mane2001]:

$$\forall t_i \in m, tw_i = \begin{cases} 1 & \text{if } t_i \text{ is in document;} \\ 0 & \text{otherwise.} \end{cases}$$

Each document vector is defined as the following:

$$\begin{bmatrix} \{1,0\}_i \\ \{1,0\}_{i+1} \\ \{1,0\}_{i+2} \\ \vdots \\ \{1,0\}_{m-l} \\ \{1,0\}_m \end{bmatrix}$$

3.3 Strict Frequency

The frequency approach builds the term weight strictly from the normalized term frequency [Mane2001].

$$\forall term_i \in m, tw_i = tf_i$$

3.4 Hadamard

The Hadamard representation is calculated as the product of the term frequency in the document and the frequency over all documents used to train the classifier.

$$\forall term_i \in m, tw_i = tf_i * df_i$$

3.5 Feature Selection

Deciding which m features or terms to use from the vector space is a problem that is solved on a per case basis, because it tends to rely heavily on the corpus. Typically one of the following methods if used to determine which features to include or exclude: document frequency, χ^2, or mutual information. The fine details of these methods are not in the scope of this paper. It is beneficial to run experiments on a subset of the corpus against each of the methods to determine an optimal solution for the corpus as a whole.

3.5.1 Document Frequency

A fairly straighforward method of feature selection leverages against the document frequency of terms in a corpus. Given that all stop words have already been removed from the term space the next step is remove terms that are singletons of are very rare in the corpus. These rare terms provide little aide in clustering documents, because they will not be significant in similarity calculations. [Mann2008], [Yang1997].

3.5.2 Mutual Information

Mutual Information measures how strong a correlation exists between a term and a class [Mann2008]. Let A be the number of times term t and c occur in the same document, B is the number of times t is without c, and C is the opposite value. Let N be the number of documents in the corpus. The MI criterion is defined as follows [Yang1997]:

$$I(t,c) = \log \frac{P_r(t \wedge c)}{P_r(t) \times P_r(c)}$$

$$I(t,c) \approx \log \frac{A \times N}{(A+C) \times (A+B)}$$

3.5.3 χ^2

χ^2 is similar to Mutual Information in that it calculates the connection between two terms t and c. The measurement is not reliable for lower frequency terms, rarer terms. Therefore it is beneficial to reduce the term space with Document Frequency before applying χ^2. Let A, B, C have the same meaning as above and let D equal the number of times neither term is in a document. Then define χ^2 as follows [Yang1997]:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

4 Document Clustering Algorithms

4.1 Centroid Based

There are multiple mechanisms to cluster documents; a common and straightforward method is the use of centroids. This is referred to as Rocchio Classification. Effectively a centroid is the average vector representing the vectors in its class. In physics it is the center of mass.

There are many variations in centroid processing that determine how documents are clustered. Some clusters are formed on nearest edge or furthest edge or closest to center. Initially all documents are each a trivial centroid. Based on the similarity of two documents they are then merged into a new centroid. Alternatively predefined exemplar centroids can be used in a semi-supervised clustering classifier. Similarity is generally calculated as the cosine similarity, see 1. How this new centroid is calculated can varies [Mann2008].

4.1.1 Cosine Similiarity

Using the vector space model, cosine similarity for documents d_i and d_j is defined as follows

[Geo2000]:

$$cos(d_i, d_j) = \frac{d_i \cdot d_j}{||d_i||_2 * \parallel d_j||_2} \quad (1)$$

If the document vectors are of unit length the cosine similarity matrix reduces to the dot product the vectors. However, centroids are not necessarily of unit length; therefore the similarity function does not reduce for those cases.

4.1.2 Average

The average method represents a centroid, C as the average vector for the collection of document vectors, S [Car2007].

$$C = \frac{1}{|S|} \sum_{d \in S} d$$

4.2 K-Nearest Neighbor

The k-nearest neighbor approach has been very successful and is also fairly straightforward. Given an input document, if the k nearest neighbors have a high enough similarity score they are grouped together (placed in the same category). High enough is defined by a minimum threshold of the similarity function used (likely cosine similarity). It is possible that some of the neighbors are very different from each other–therefore we merge with the nearest to the input document. In the event that multiple neighbors are similar to each other (or possibly in the same category–in supervised classification) then the similarity is calculated against their cumulative score. Given a document $\vec{d_i}$ with respect to a category c_j and a threshold b_j the decision rule is [Yang1999]:

$$y(\vec{d_i}, c_j) = \sum_{\vec{d_i} \in kNN} sim(\vec{x}, \vec{d_i}) y(\vec{d_i}, c_j) - b_j$$

$y(\vec{d_i}, c_j) \in \{0, 1\}$ is the classification; either 1 or 0. A threshold is used because a document may fit in more than one category. A simpler problem assumes that a document can only be assigned to one category; however this can prove slightly inaccurate in information retrieval scenarios [Yang1999].

4.3 Support Vector Machines

Support Vector Machines are state-of-the-art classifiers which attempt to divide the documents on a boundary in hyperspace. The following relates more to supervised classification where classes are predefined. SVMs view documents in the vector space model. The boundary SVMs find it the decision surface that is maximally far away from any vector. This decision surface attempts to seperate the data points into classes. The margin is defined as the distance between parallel points [Mann2008] [Yang1999]. Support vectors are the data points that define the position of the seperator. Given a weight vector $\vec{w}$, which is the decision hyperplane perpendicular normal vector and an intercept term b. Let $\vec{x}$ be a vector to classify, then all points satisfy [Mann2008]:

$$\vec{w} \cdot \vec{x} - b = 0$$

Let $\mathbb{D} = \{(\vec{x}_i, y_i)\}$, where $\vec{x}_i$ is a point in the training set and y_i is a classification of either +1 or -1 (for the 2-class example). The SVM problem is defined as finding $\vec{w}$ and b that satisfy the following:

$$\vec{w} \cdot \vec{x}_i - b \geq +1 \quad for \quad y_i = +1$$
$$\vec{w} \cdot \vec{x}_i - b \leq -1 \quad for \quad y_i = -1$$

The SVM problem can be solved with quadratic programming techniques [Yang1999].

4.4 Naïve Bayes

The Naïve Bayes classifier follows Bayes theorem. Given a set of terms, the probability of them appearing in a specific class is based on all the conditional probability of any of the terms occurring. Naïve Bayes simplifies this by stating that all term probabilities are independent [Yang1999]. Without the assertion of independence the probability calculations would complicate exponentially as terms are added to the feature set. Given a document d, a class c, a term t_k, and n_d is the term count for d, the probability is defined as:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

4.5 N-gram Clustering

Given a corpus, tokens can be created as n-sized pieces of data known as n-grams. These tokens can also be used to represent the term space in a beneficial way for clustering. Given a corpus, the documents with similar n-grams can be clustered together in a similar methods as with normal tokens. This clustering can actually be applied in a language independent way as demonstrated by Damashek. The document clusters will be by topic and by language.

Documents in the same language with the same category will cluster together. Interestingly enough it is quite likely that documents of the same category in similar languages will cluster together as well. This approach can also be applied to determine the language of a document. The clustering will also separate the documents into clusters/categories by language [Dama1995].

4.6 Neural Networks

Neural networks have been used to classify documents with the usage of a training set (supervised classification) with success. One drawback to utilizing a neural network for classification is the considerably higher training cost involved [Yang1999].

5 Future Research

5.1 Unknown Malware Detection via Clustering Algorithms

In the last ten years there has been a push for detecting new malware via heursitics. However, these require executing the code natively or in an emulator. Other malware detection systems use signatures that look for instruction sequences of known virii. Static analysis of binary code to create a feature set cover to detect commonality in malware has been explored. This utilizes information retrieval techniques such a n-grams [Arno2000]. A training set of known malware is broken into n-grams as a binary document. Arnold et al, conducted virus detection research using disjoint feature sets and a voting neural network system. The voting system reduced false positives because of the requirement that the input binary had to be flagged by more than one classifier. In a binary document if the n-gram is too small there is a higher likelihood of it appearing in many benign sequences.

With feature based classifiers too many features can reduce accuracy. Therefore measures need to be taken to reduce the feature space when using classifiers to detect malicious software. One important filter for feature space is looking at the document frequency of each feature. Choosing only those features that are infrequent (simiarly avoiding "stop words"). These features can then be used with document classifiers to detect malware [Mosk2008]. Research into the field of static binary analysis is still rather active because of the need to reduce false positives, and correctly ignore interleaved benign code sequences, which if they are done correctly can break n-grams into features with no value.

5.2 Clusting Queries to Remove Redundency and Improve Recall

Give some set of queries Q. Each query q_i in Q can be clustered as if it were a document (because effectively it is a document–just a short one). Given these new clusters of queries when a user inputs a query into the system the new query is run through the classifier to determine which cluster is nearest in the set of clusters C. Once the new query x_i is identified as belong to cluster $c_i \in C$; all the other queries in C can also be run and their results joined as a set.

Another application is linguistic semantic analysis. Given two different queries that return the same documents may very well have the same information need and be semantically identical. An example of this is "automatica translation" and "machine translation." A query processing system with a thesaurus may appropriately handle the above query, however a thesaurus system typically adds terms to a query. The speedup is gained partially when the query is identified as one that is already solved and the cached results can be returned. The other gain is from the relevance correlation of the results from the semantically identical queries.

6 Conclusion

To determine the best solution for the corpus it is a good practice to build a small scored subset corpus and run experiments with different document represetations, feature selection algorithms and classifiers. It is important to note that aggresive feature selection may have a negative impact on the term space. Even some trivial terms may provide a certain amount of information regarding the subject matter of the input document [Joac1997].

Centroid classifiers summarize the characteristics of a class. A centroid is the average document vector for a class and therefore will have all the key features or terms of the class. Another advantage of this characterization of a class with centroids is that prevalent features need not all be present in each document. Centroids can provide better results than Naïve Bayesian because the occurence of terms in a corpus is not independent necessarily independent. However, Naïve Bayesian classifiers are still prominent in the field because of their effectiveness in classifying documents.[Geo2000].

Support Vector Machines have been proven to work well in higher dimensional feature sapces, and do not require parameter tuning.[Joac1997].

Experiments have indicated that Document Frequency and χ^2 are strongly correlated. Therefore it may be feasible to use strictly Document Frequency to reduce term space for cases where the classifier is slow or complex and would benefit from a sparser term space [Yang1997].

References

[Mane2001] L. Manevitz and M. Yousef, "One-Class SVMs for Document Classification," *Journal of Machine Learning Research 2*, pp. 139-154, Decc 2001

[Car2007] A. Cardoso-Cachopho and A. Oliveira, "Semi-supervised Single-label Text Categorization using Centroid-based Classifiers," 2007

[Geo2000] E. Han and G. Karypis, "Centroid-Based Document Classification: Analysis and Experimental Results," *4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pp. 424-431, 2000

[Mann2008] C. Manning and P. Raghavan and H. Schuetze, "Introduction to Information Retrieval," 2008

[Yang1997] Y. Yang and J. Pedersen, "A Comparative Study on Feature Selection in Text Categorization," *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 412-420, 1997

[Yang1999] Y. Yang and X. Liu, "A re-examination of text categorization methods," *ACM SIGIR*, 1999

[Mosk2008] R. Moskovitch, D. Stopel, C. Feher, N. Nissim, Y. Elovici, "Unknown Malcode Detection via Text Categorization and the Imbalance Problem," *IEEE International Conference on Intelligence and Security Informatics*, pp. 156-161, 2008

[Arno2000] W. Arnold and G. Tesauro, "Automatically Generated WIN32 Heuristic Virus Detection," *Virus Bulletin Conference*, Sept. 2000

[Joac1997] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," 1997

[Dama1995] M. Damashek, "Gauging Similarity with n-Grams: Languages-Independent Categorization of Text," *Science*, pp 843-848, Feb. 1995