A New Method for Bootstrapping an Automatic Text Classification System Utilizing Public Library Resources

Arash Joorabchi[1] and Abdulhussain E. Mahdi [1]

[1] Department of Electronic and Computer Engineering, University of Limerick, Ireland
{Arash.Joorabchi, Hussain.Mahdi}@ul.ie

Abstract. One of the major obstacles in using Machine Learning (ML) based Automatic Text Classification (ATC) methods in real-world applications is that they require a large corpus of manually classified documents for training the classification algorithm. Manual labeling of documents is an expensive and time consuming task which is a not feasible in many cases. This paper proposes a new flexible ML-based ATC system which uses a novel bootstrapping technique to automatically collect and build an appropriately labelled training corpus. In specific, it describes the development of a bootstrapping module which utilises public library resources and publicly available information on published books for automatic building of a labelled corpus. This corpus is then used by the system to train an optimised naive Bayes and a linear SVM document classification algorithm. Preliminary experimental results for evaluating the system's performance are presented and discussed.

1 Introduction

Automated Text Classification/Categorization (ATC) is the automatic assignment of natural language text documents to one or more predefined categories or classes according to their contents. ATC has been an active field of research for four decades. In recent years, due to the explosive growth in the number of electronic documents available on the Internet, intranets, and digital libraries, there has been a growing need for automatic systems capable of indexing and managing such large volumes of data. In response to this need, ATC research has attracted increasing interest from researchers in artificial intelligence, information retrieval, and data management communities. Examples of text classification applications include classification of Web pages, sorting electronic mail or news articles, and learning user reading interests, to name a few.

Until the late '80s, the use of rule-based methods was the dominant approach to ATC. Rule-based classifiers are built by knowledge engineers who inspect a corpus of labelled sample documents and define a set of rules which are used for identifying the class of unlabelled documents. In the '90s, with the advances in the area of Machine Learning (ML) and the emergence of high processing computers capable of analysing thousands of documents in a reasonably short time, ML algorithms were widely adopted in ATC and superseded the rule-based methods. In the latter approach, the ML algorithms play the role of knowledge engineers in corresponding rule-based

systems and automatically learn the distinguishing characteristics of each class by analysing a collection of sample documents in that class. Since then, almost all of the general ML algorithms, such as naive Bayes, k-nearest neighbor, support vector machines, artificial neural networks, decision trees and genetic algorithms have been applied to this problem with considerable success, and currently support vector machines (SVMs) [1] are widely known to yield the best results in terms of accuracy.

There are many parameters that affect the accuracy of an ML-based ATC system. These include classification scheme (e.g., flat vs. hierarchical), domain of documents (e.g., wire news vs. scientific papers), quantity and quality of training documents, document frequency distribution across categories in the training corpus (i.e., problem of skewed datasets), document representation method (e.g., bag-of-words vs. bag-of-phrases), term weighting mechanism (e.g., binary vs. multinomial), feature reduction techniques (e.g., word stemming, and stop word removal), feature selection method (e.g., Document Frequency vs. Information Gain), and obviously the type of classification algorithm used (e.g., naive Bayes vs. SVM). Different combinations of these parameters could yield different results in terms of classification performance measures. Due to the diversity of parameters involved, it is difficult to compare the results of many proposed methods. Another factor contributing to this problem is that different datasets are used in different studies, or the same datasets are often splitted differently between training and test subsets. Despite these issues, ML-based ATC methods have shown great potential as demonstrated by reported experimental results [2]. On the other hand, the performance of these methods in real-world operational projects is considerably below that demonstrated in the experimental setting [3, 4]. This problem has been mainly associated with the lack of high-quantity and/or -quality datasets for training the ML algorithms. Labeling sample documents for the purpose of building a training corpus is a tedious and expensive process which usually requires the expertise of professional cataloguers. This issue is known in literature as the bootstrapping problem. Motivated by this problem, semi-supervised approach for text classification has been introduced, aiming to reduce the required number of manually labelled sample documents by using different strategies for incorporating large easily-obtained unlabelled data (e.g., see [5-7]). In this paper we describe a new bootstrapping technique for ML-based ATC systems which does not require any manually labelled sample documents. The proposed technique is based on utilising public library resources (i.e., standard library classification schemes, and online public access library catalogues) and book description information retrieved from online book sellers' websites.

The rest of the paper is organised as follows: Section 2 discusses the three major components of an ML-based ATC system and elaborates on our choice for each component. Section 3 describes the developed document classifier system and its various components in details. Section 4 describes the evaluation process carried out to assess the performance of the system, presenting and discussing some preliminary and experimental results. Section 5 concludes the paper and discusses future work.

2 ML-based ATC System Components

Our main goal in this work is to develop a dynamic ML-based ATC system which can be adopted for a wide range of classification tasks with a minimum amount of effort required. In this system, users only provide the name of classes, and a bootstrapping technique is deployed to automatically build a dataset for training the classification algorithm accordingly. We have identified three main components that are necessary for achieving this goal, namely, a universal classification scheme, a high quality training corpus, and a suitable classification algorithm. The following subsections describe each of these components in details.

2.1 Universal Classification Scheme

In order to allow the users to create a tailored classification scheme suitable for the requirements of their document classification task, we need to adopt a comprehensive universal classification scheme covering almost all subjects of human interest. The universal classification scheme of choice will act as a pool of categories/classes that can be selectively adopted by the users to create their own classification scheme according to their classification task. In an investigation into universal classification schemes, we came across the library classification systems as a suitable choice for our purposes.

A library classification is a system of coding and organizing library materials according to their subjects that simplifies subject browsing. Library classification systems have been used by cataloguers to classify books and other materials (e.g., serials, audiovisual materials, computer files, maps, manuscripts, and realia) in libraries for over a century. The two major library classification systems, Dewey Decimal Classification (DDC) [8] and Library of Congress Classification (LCC) [9] , have undergone numerous revisions and updates over this time. DDC and LCC are the most widely used universal classification schemes in libraries today and they cover nearly all subjects of interest to human. There are approximately 100,000 different classes in LCC and the class number of DDC is not far from it. LCC scheme allows for greater precision in most fields, however, for the purpose of the reported work we adopted DDC for two main reasons:

1. DDC is used for classifying items in about 80% of libraries around the world and therefore the number of manually classified items according to DDC is much grater than to LCC. This makes DDC a better choice for our bootstrapping algorithm which is based on utilising the resources that have been manually classified according to a universal classification scheme (see 2.2 for details).
2. DDC has a fully hierarchical structure while LCC is not hierarchical and usually leans toward alphabetic or geographic sub-arrangements. The hierarchical relationships among topics in DDC are shown by classification numbers that can be continuously subdivided. The hierarchical feature of DDC allows the development of effective GUI interfaces that enable users to easily browse and navigate the scheme and quickly find the categories that they are interested in without requiring knowledge of the classification scheme or its notational

representation. [10] Explains the basic functions of browsing and searching that need to be supported in relation to a library classification scheme.

2.2 Training Corpus

After deciding on DDC as the universal classification scheme of choice, we needed to find a source from which we could automatically collect manually labelled text documents that are classified according to DDC. These documents are needed for creating a dataset which is used for training the ML-based classification algorithm. Considering the number of classes in DDC (≈100,000 classes), the manual labeling of sample documents is infeasible in this case, as even providing a single sample document in each class requires manual labeling of about 100,000 documents. Faced with this problem, a number of researchers have used an alternative approach to untilising library classification systems for ATC which is not ML-based. This approach comprises string-to-string matching between words in a term list extracted from library thesauruses and classification schemes, and words in the text to be classified (see [11] for a review of these works). However, the accuracy of this approach is considerably lower than the ML-based approach [12].

DDC scheme is used on daily basis by thousands of expert cataloguers in libraries around the world to classify books and other text items. These enormous collections of classified text can potentially be used for building a very high quality training corpus for automatic ML-based classification of text documents according to DDC scheme. However, this is practically impossible as the textual contents of library items are not electronically available and/or are copyrighted. In this work, we use the small parts of books such as cover pages which are publicly available on online books sellers' websites as an alternative to the full text of classified books. In specific, we use the textual content of *editorial reviews* section of book descriptions provided by the online bookseller Amazon[1]. This section contains short descriptions of the book such as editors' reviews, topics covered, and the back cover. Although the editorial reviews texts are usually short(less than 500 words) and are not available for all the books, they contain valuable keywords which expose the books' main topics/categories. The bootstrapping component of our classification system builds a corpus of training documents for each class by retrieving the editorial reviews section of books which belong to that class (see 3 for details).

2.3 Classification algorithm

Naive Bayes (NB) and Support Vector Machines (SVM) are two of the most popular machine learning algorithms applied to the problem of ATC to date. NB is widely used in real-world applications such as spam filtering because it is fast and easy to implement. SVM is more complex than NB and slower to train but it is known as the state-of-art algorithm for ATC problem. In this work we experiment with both of these algorithms. We have implemented an optimized version of naive Bayes called

[1] http://www.amazon.com

Transformed Weight-normalized Complement Naive Bayes (TWCNB) as described in [13]. The corrections applied to NB in TWCNB result in a fast algorithm that is competitive with SVM. We also experiment with a linear SVM classifier called LIBLINEAR [14] which is an optimised implementation of SVM suitable for large linear classification tasks with thousands of features, such as ATC.

3 ATC System Description

In this section we describe a prototype ATC system which has been developed for testing and evaluating the performance of the proposed bootstrapping method. The prototype is depicted in Fig.1, where the main processing stages and components of the system are illustrated.

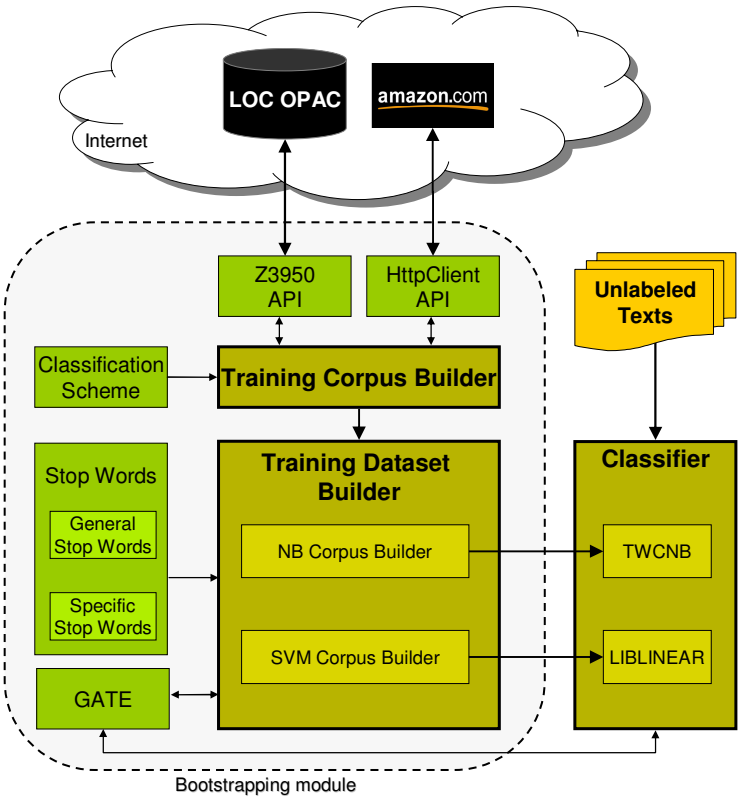

Figure 1. Overview of the prototype ATC system.

The rest of this section describes the design, implementation and operation of the three main components of the system comprising a Training Corpus Builder, a Training Dataset Builder, and a Classifier.

3.1 Training Corpus Builder

The task of Training Corpus Builder (TCB) component is to create a corpus of labelled sample documents according to the user's classification scheme. The classification scheme is an XML file containing a set of classes chosen from the DDC scheme. Each class element has a caption (i.e., class name) and a Dewey number (i.e., class id) attribute according to DDC scheme. For example the Dewey number 796.357 corresponds to the *Baseball* class which is one of the descendents of *Arts & recreation* main class as shown in table 1.

Table 1. Baseball class in DDC scheme

Dewey No.	Caption
700	Arts & recreation
790	Sports, games & entertainment
796	Athletic and outdoor sports and games
796.3	Ball games
796.35	Ball driven by club, mallet, bat
796.357	**Baseball**

The TCB component reads the user's classification scheme and creates a list of classes that it contains. The Z3950 protocol [15] is then used to search the Online Public Access Catalogue (OPAC) of US Library of Congress (LOC)[2] for all the items that have been labelled with the Dewey numbers in the list. The returned search results contain the records of matching catalogued items in MARC21 format [16]. Each record holds bibliographic and classification information about an item including a unique universal identifier (i.e., ISBN number) for book items. The TCB component iterates through the retrieved records and extracts the ISBNs of all the books and divides them into different groups according to their corresponding Dewey number. This process results in sets of ISBNs grouped according to Dewey numbers. For example, an ISBN set labelled with Dewey number 796.357 would contain a list of ISBNs of the books that baseball game is their main subject (judged by expert cataloguers in LOC). The TCB component iterates through these lists and uses the collected ISBNs to retrieve the book description web pages of the corresponding books from the Amazon bookseller website. On Amazon's USA-targeted web site, a book's web address (URL) always contains its ISBN in the format of *http://amazon.com/gp/product/ISBN-VALUE*, which enables easy and direct access to the book description web pages, provided targeted books' ISBNs are available. Finally retrieved book description web pages from Amazon grouped according to the corresponding Dewey number of their ISBNs are passed to the Training Dataset Builder component for creating training datasets.

[2] http://catalog.loc.gov/

3.2 Training Dataset Builder

The main task of Training Dataset Builder (TDB) component is to create two different Training Datasets, one for training the TWCNB classifier (naïve Bayes-based) and another for training the LIBLINEAR classifier (SVM-based) using the book description web pages collected by the TCB component. The reason for creating two training datasets is that TWCNB and LIBLINEAR classifiers have different training mechanisms and dataset requirements. Before starting the process of building the datasets, TDB component first parses out the *editorial reviews* sections of the collected book description pages, removes the headings that appear in all editorial sections, and converts the results from HTML to pure text format. The resulting texts of this filtering process are ready to be used for building the training datasets.

The original naïve Bayes text classification algorithm, as explained by Mitchell in [17], uses a Bag-Of-Words (BOW) model for representing the datasets. In this model, a text such as a document is represented as an unordered set of unique words and their associated frequency counts found in the text. In order to create a dataset for training the naïve Bayes all the documents that belong to the same class are concatenated to create a single large document and the BOW representation of these documents are used for training the algorithm. The TWCNB algorithm, used in this work, is an optimized version of the original naïve Bayes which applies a number of transformations to the BOW model to better align the model and the data (see [13] for details of these transformations).

LIBLINEAR also uses the Bag of words (a.k.a. Vector of Words) model for representing the texts in the training dataset. However, in the dataset used for training the LIBLINEAR, each sample document represented in BOW model is a training instance on its own, as compared to the naïve Bayes training dataset in which all texts belonging to the same class are merged to create a single large document. Also LIBLINEAR like the other SVM implementations does not accept words as attributes, therefore, each word in the bag of words representation of the document must be replaced by a numeric value.

In order to build two training datasets with above specifications from the extracted editorial reviews texts, the TDB component uses the Gate toolkit [18] to tokenize each text and extract its words, stem the extracted words to replace them with their root, and remove the stop words to avoid the words which appear in almost all of the documents and therefore are irrelevant to the class of documents. TDB component uses a generic stoplist of 526 words from Bow toolkit [19], which contains common words such as "the", "of", "is", etc, and enhances it with a domain-specific stoplist of 50 words, such as “book”, “author”, “title” which appear in majority of extracted book editorial reviews texts.

3.3 Classifier

The task of the Classifier component of the system is to automatically classify the incoming unlabelled documents according to the classification scheme provided by the user and assign a Dewey number to each document. The incoming documents are converted to pure text format, tokenized, and their words are stemmed using the Gate

toolkit. The texts resulting from this process are then converted to BOW representation model and fed to TWCNB and LIBLINEAR classifiers for classification. In this paper we do not go into the details of naïve Bayes and SVM text classification algorithms which TWCNB and LIBLINEAR classifiers are based on, as they both have been extensively studied in the literature, e.g., see [1, 13, 20-22].

4 System Evaluation and Experimental Results

We have evaluated the accuracy of the developed system with a standard benchmarking dataset for text categorization called 20news-18828 collection which is a refined version of the well-known 20-Newsgroup dataset [23]. It is a collection of 18,828 Usenet newsgroup articles, partitioned (nearly) evenly across 20 different newsgroups. In order to use this collection as the test dataset for evaluating our system, we mapped eight classes in 20-Newsgroup to their corresponding classes in Dewey Decimal Classification scheme, built an XML classification scheme accordingly, and ran the bootstrapping module of the system to automatically collect and build two datasets for training the two classifiers of the system. Table 2 shows the mapping of newsgroups to DDC and also the number of texts that the bootstrapper component has collected for each class.

Table 2. 20-Newsgroup to DDC mapping

newsgroup	*Dewey Number*	*Dewey Caption*	*No. of training texts collected*
sci.space	520	Astronomy and allied sciences	810
rec.sport.baseball	796.357	Baseball	997
rec.autos	796.7	Driving motor vehicles	587
rec.motorcycles	796.7	Driving motor vehicles	587
soc.religion.christian	230	Christian theology	1043
sci.electronics	537	Electricity and electronics	713
rec.sport.hockey	796.962	Ice hockey	270
sci.med	610	Medicine and health	1653

The remaining newsgroups, such as *miscellaneous.forsale*, *talk.politics.guns*, and *talk.religion.miscellaneous*, did not correspond with any class in DDC. For assessing the performance of the ATC system, we used the precision measure, which is computed as follows for each class:

$$\text{Precision} = \frac{\text{Total number of correctly classified documents}}{\text{Total number of classified documents}} . \quad (1)$$

First, to ensure the accuracy of the in-house implemented TWCNB and externally-developed LIBLINEAR classifiers for our automatic document classification task, in

a preliminary experiment, we used the whole 20news-18828 collection to measure the accuracy of these two classifiers and compare it with the results of similar experiments reported in the literature. In this experiment we have spilited the dataset into 80% training data and 20% testing data per class. The micro-average precision (global calculation of precision regardless of topics) achieved for TWCNB and LIBLINEAR are 0.8504 and 0.8596 respectively which closely match those reported in [13] and other similar experiments. After ensuring the integrity of the classifiers, we measured the accuracy of the developed system in an experiment in which ≈8,000 articles from eight newsgroups in table 2 (≈1000 articles per class) were classified by the system. Table 3 shows the achieved percentage of precision in each class by the TWCNB classifier. The average precision percentage achieved by TWCNB is 77.87%. It is not possible to properly compare the accuracy of our system with those achieved by the other bootstrapping methods proposed in the literature, as all the methods that we know of, unlike our system, require at least a limited number of labelled texts or keywords per class to initiate the bootstrapping process, and they also use different classification schemes and test datasets. However, the achieved accuracy seems to be quite competitive with those achieved using semi-supervised bootstrapping methods for ACT reported in literature, e.g., see [5-7].

Table 3. TWCNB classification accuracy results

newsgroup	***Precision%***
sci.space	69.19
rec.sport.baseball	96.78
rec.autos	74.74
rec.motorcycles	71.02
soc.religion.christian	89.36
sci.electronics	69.92
rec.sport.hockey	75.77
sci.med	76.23
	Avg. 77.87

To our surprise, the LIBLINEAR classifier with achieved average precision of 68% turned out to be considerably less accurate than TWCNB for our classification task. The reason for under performance of the linear SVM classifier in the specific setting of our system needs further investigation. However, these results again demonstrate the fact that the best choice of classification algorithm highly depends on the specific features of the training and test datasets of the classification task and there is no optimal classification algorithm for all ATC scenarios, as also stated by other researchers [2, 24].

5 Conclusion and Future Work

In this paper, we highlighted the lack of sufficient training data as one of the biggest practical challenges in fielding a machine-learning classifier in real-world ATC applications. We proposed a new fully automated bootstrapping method which automatically collects and builds a training dataset based on the user's classification scheme and evaluated its accuracy by developing and testing a prototype ATC system. The training datasets built by the bootstrapping mechanism were used to train a naïve Bayes-based and a SVM-based classifier and their accuracy was measured in an experiment in which they were used to classify ≈8,000 newsgroup articles to eight classes. The accuracy achieved by the new proposed bootstrapping method is quite competitive with those achieved by semi-supervised learning methods. However, we believe the proposed bootstrapping method has the potential of achieving much higher accuracies, given richer sources for collecting and building the training dataset. In order to further investigate and exploit this potential, in future we plan to experiment with alternative sources for collecting and building the training dataset. One of these sources which seems very promising is Google Book Search project[3]. It is a tool from Google that searches the full text of books that Google and its library partners scan, OCR, and store in its digital database. Google does not reveal how many books they have already scanned to date. However, As of March 2007, The New York Times[4] reported that Google has already digitised one million books and they are scanning more than 3,000 books per day, a rate that translates into more than 1 million annually. Google does not provide public access to the contextual content of these books due to copyright restrictions. However, each book includes an *'About this book'* page which provides a short description of the book and a set of links to WebPages that review the book and describe its content. Most importantly, it contains a list of key terms and phrases that are statistically significant in the book. The majority of these key terms are domain-specific, semantically rich, and directly related to the core subject of the book and, therefore, they could substantially increase the quality of training corpus produced by our bootstrapping algorithm and boost its accuracy. To conclude, we believe that the enormous intellectual effort put into creating and maintaining library classification systems and classifying tens of millions of library items over the last hundred years has a lot of potential to be leveraged in the field of ATC which is yet to be explored to its fullest.

References

[1] Joachims, T. Text Categorization with Suport Vector Machines: Learning with Many Relevant Features. In *Proceedings of the 10th European Conference on Machine Learning* (Dorint-Parkhotel, Chemnitz, Germany, 1998). Springer-Verlag. [Online]. Available: www.cs.cornell.edu/people/tj/publications/joachims_98a.ps.gz

[2] Sebastiani, F. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34, 1 (2002), 1-47. DOI: http://doi.acm.org/10.1145/505282.505283

[3] http://books.google.com/

[4] http://www.nytimes.com/2007/03/10/business/yourmoney/11archive.html

[3] Lewis, D. D. and Sebastiani, F. Report on the Workshop on Operational Text Classification Systems(OTC-01). *ACM SIGIR Forum newsletter* 35, 2 (2001). DOI: http://www.sigir.org/forum/F2001/textClassification.pdf

[4] Dumais, S. T., Lewis, D. D. and Sebastiani, F. Report on the Workshop on Operational Text Classification Systems (OTC-02). *ACM SIGIR Forum newsletter* 36, 2 (2002). DOI: http://www.sigir.org/forum/F2002/sebastiani.pdf

[5] Nigam, A. M. a. K. Text Classification by Bootstrapping with Keywords. *A. McCallum, K. Nigam, Text Classification by Bootstrapping with Keywords, EM and Shrinkage, ACL Workshop for Unsupervised Learning in Natural Language Processing, 1999*(1999). DOI: citeseer.ist.psu.edu/article/mccallum99text.html

[6] Adami, G., Avesani, P. and Sona, D. Clustering documents into a web directory for bootstrapping a supervised classification. *Data & Knowledge Engineering*, 54, 3 (2005), 301-325. DOI: http://www.sciencedirect.com/science/article/B6TYX-4F01KFB-1/1/2c5540f69632f4d17c85d3ae43eb64e0

[7] Guzman, R., Montes, M., Rosso, P. and Villasenor, L. Improving Text Classification by Web Corpora. In *Proceedings of the 5th AtlanticWeb Intelligence Conference (AWIC'2007)* (Fontainbleau, France, June 25–27, 2007). [Online]. Available: http://www.dsic.upv.es/~prosso/resources/GuzmanEtAl_AWIC07.pdf

[8] Dewey, M. *Dewey Decimal Classification (DDC)*. (OCLC Online Computer Library Center, 1876) [cited 2008 January]; [Online]. Available: http://www.oclc.org/about/default.htm

[9] Putnam, H. *Library of Congress Classification (LCC)*. (Library of Congress, Cataloging Policy and Support Office, 1897) [cited 2008 January]; [Online]. Available: http://www.loc.gov/catdir/cpso/lcc.html

[10] Slavic, A. Interface to classification: some objectives and options. *Extensions and Corrections to the UDC*, No. 28(2006). DOI: http://dlist.sir.arizona.edu/1621/

[11] Golub, K. Automated subject classification of textual Web pages, based on a controlled vocabulary: Challenges and recommendations. *New Review of Hypermedia and Multimedia*, 12, 1 (2006), 11 - 27. DOI: http://www.informaworld.com/10.1080/13614560600774313

[12] Golub, K., Ardö, A., Mladenić, D. and Grobelnik, M., *Comparing and Combining Two Approaches to Automated Subject Classification of Text*, in *Research and Advanced Technology for Digital Libraries*. 2006. p. 467-470. http://dx.doi.org/10.1007/11863878_45

[13] Jason D. M. Rennie, L. S., Jaime Teevan, David R. Karger. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. In *Proceedings of the Proceedings of the Twentieth International Conference on Machine Learning (ICML)* (Washington, DC USA, 2003). DOI: http://people.csail.mit.edu/jrennie/papers/

[14] Lin, C.-J., Wang, X.-R., Chang, K.-W., Hsieh, C.-J. and Fan, R.-E. *LIBLINEAR:a library for large linear classification* (Machine Learning Group at National Taiwan University 2008) [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[15] *Z39.50*. (International Standard Maintenance Agency - Library of Congress Network Development and MARC Standards Office, October 1, 2007) [cited 2008 January]; [Online]. Available: http://www.loc.gov/z3950/agency/

[16] *MARC standards*. (Library of Congress Network Development and MARC Standards Office, December 5, 2007) [cited 2008 January]; [Online]. Available: http://www.loc.gov/marc/

[17] Mitchell, T., *Machine Learning*. 1997, McGraw-Hill. p. 180-184.

[18] Cunningham, H., Maynard, D., Bontcheva, K. and Tablan, V. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)* (Philadelphia, US., July, 2002). [Online]. Available: http://gate.ac.uk/gate/doc/papers.html

[19] McCallum, A. *Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering*. (Released under the open source LGPL licence, 1998) [Online]. Available: http://www.cs.cmu.edu/~mccallum/bow/

[20] McCallum, A. and Nigam, K. A comparison of event models for Naive Bayes text classification. In *Proceedings of the AAAI-98 Workshop on "Learning for Text Categorization"* (Wisconsin, USA, 1998). [Online]. Available: http://www.cs.umass.edu/~mccallum/papers/multinomial-aaai98w.ps

[21] Kibriya, A., Frank, E., Pfahringer, B. and Holmes, G., *Multinomial Naive Bayes for Text Categorization Revisited*, in *AI 2004: Advances in Artificial Intelligence*. 2005. p. 488-499. http://www.springerlink.com/content/gmpy00w0be5la9k0

[22] Joachims, T. *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*. Kluwer Academic Publishers, 2002.

[23] Lang, K. *The 20 Newsgroups data set, version 20news-18828*. (1995) [Online]. Available: http://people.csail.mit.edu/jrennie/20Newsgroups/

[24] Manning, C. D., Raghavan, P. and Schütze, H., *Support vector machines and machine learning on documents*, in *Introduction to Information Retrieval*. 2008, Cambridge University Press. p. 293-320. http://www-csli.stanford.edu/~hinrich/information-retrieval-book.html