Categorizing Gigabytes: Experiments on the RCV1 Corpus

Domonkos Tikk[1], Zoltán Bánsághi[1], György Biró[2]

[1] Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Magyar tudósok krt. 2, H-1117 Budapest, Hungary
tikk@tmit.bme.hu, empzoooli@gmail.com

[2] Textminer Ltd., Gyulai P. u. 37, H-1029 Budapest, Hungary
george.biro@gmail.com

Abstract: This paper presents categorization results performed by means of HITEC categorizer tool on the new benchmark document collection of text categorization, the Reuters Corpus Volume 1 (RCV1). RCV1 is an archive of over 800,000 manually categorized newswire stories made available by Reuters in 2000 for research purposes. This collection was released to take place of the Reuters-21578 collection that has been used widespread in the text retrieval community. This paper inted to add some interesting result to the characterization of RCV1 and HITEC categorizer.

1 Introduction

Nowadays the immense and exponentially growth in the number of electronic documents stored on the internet, corporate intranets and data warehouses brings internsive needs for *text mining* softwares that are able to efficiently index, handle, search and categorize (textual) data of in large quantity. These tasks require – due to the quantity of data – automatized methods document handling, as manually it is no longer amenable in that size, requiring a vast amount of time and cost.

One of the most significant task of text mining softwares is text categorization (TC). Until recently, in the literature of TC much attention has been given to flat categorizers that are able to classify documents into unstructured category systems. However, as the number of topics becomes larger, flat categorizers face the problem of complexity that may incur rapid increase of time and storage, and compromise the perspicuity of categorized subject domain. A common way to manage complexity is using a hierarchy, and text is no exception. Hierarchic category systems (also called *taxonomy*) offers straighforward way to find and browse data at arbitrary refinement. Before the release of RCV1 there was no available widely used benchmark text collection for hierarchic TC, and therefore researchers performed their tests on diverse corpora (see e.g. [1, 2, 3, 4, 5, 6, 7, 8]) that made the comparison of the methods extremely difficult and unreliable [9]. Therefore it is a straighforward step for developers of hierarchical categorizers to run tests on new collection.

2 RCV1 Corpus

The RCV1 corpus is presented here based on the papers [10, 11].

Reuters is the largest international text and television news agency. Its editorial division produces some 11,000 stories a day in 23 languages. Stories are made available via online databases and other archival products. RCV1 is drawn from one of the online databases of Reuters. It was intended to consist of all and only English language stories produced by their journalists between August 20, 1996, and August 19, 1997. The data is availabe on two CD-ROMs and has been formatted in XML, that went through formation, verification and validation before the release.

The stories cover the range of content typical of a large English language international newswire. They vary from a few hundred to several thousand words in length.

The documents are indexed by category codes from three sets (Topics, Industries, and Regions). Topic codes were assigned to capture the major subjects of a story. They were organized in four hierarchical groups: CCAT (Corporate/Industrial), ECAT (Economics), GCAT (Government/Social), and MCAT (Markets). This code set provides a good example of how controlled vocabulary schemes represent a particular perspective on a data set. The RCV1 articles span a broad range of content, but the code set only emphasizes distinctions relevant to Reuters customers. For instance, there are three different Topic codes for corporate ownership changes, but all of science and technology is a single category (GSCI).

Originally there are 126 Topic codes, among which only 103 were assigned to stories and which are therefore recommended to be used in TC experiments. All of these 103 codes occur at least once in the RCV1 dataset. The corpus frequencies span five orders of magnitude, from 5 occurrences for GMIL (Millennium issues), to 381,327 for CCAT (Corporate/industrial). In our experiments we only used the topic code categories.

Industry codes were assigned based on types of businesses discussed in the story. They were grouped in 10 sub-hierarchies, such as I2 (Metals and minerals) and I5 (Construction). The Industry codes make up the largest of the three code sets, supporting many fine distinctions.

Region codes included both geographic locations and economic/political groupings. No hierarchical taxonomy was defined.

3 On the Categorizer

We have developed since 2001 a categorizer algorithm called UFEX (Universal Feature Extractor) [12, 6, 7], and its implementation that is referred to as HITEC, which stands for **HI**erarchical **TE**xt **C**ategorizer.

3.1 The Engine of UFEX

UFEX method aims at determining relevant characteristics of a set of categories based on training entities. It is particularly optimized to handle hierarchically organized category structures. The nature of the training entities is independent from UFEX as it applies an internal representation form, therefore it is able to work on arbitrary kind of data (e.g. text, image, numerical measurements) that can be described by numerical vectors of features. The basec idea of UFEX is described in details in [13]. For simplicity, in the next we will use the TC-specific notations. Here we remark again that, nevertheless, UFEX is designed to be able to process arbitrary numberical data.

The core idea of UFEX is an iterative learning module that gradually trains the classifier to recognize constitutive characteristics of categories and hence to discriminate typical documents belonging to different categories.

Characteristics of categories are captured by typical terms occurring frequently in documents of the corresponding categories. We represent categories by weight vectors, called *category descriptors* (or simply descriptors), where an element of this vector refers importance of a term (typically word) discriminating the given category from others. The training algorithm of UFEX sets and maintains category descriptors in a way that allows the classifier to be able to categorize documents with high accuracy in the appropriate category. The training starts with zero descriptors.

We now briefly describe the training procedure. First, when classifying a training document we compare it with category descriptors and assign the document to the category of the most similar descriptor. When this procedure fails finding correct category we raise the weight of such features in category descriptors that appear also in the given document. If a document is assigned to a category incorrectly, we lower the weight of such features in descriptors that appear in the document. We tune category descriptors by finding the optimal weights for each feature in each category descriptor by this awarding-penalizing method. The training algorithm is executed iteratively and ends when the performance of the classifier cannot be further improved significantly. See the block diagram of Figure 1 for an overview.

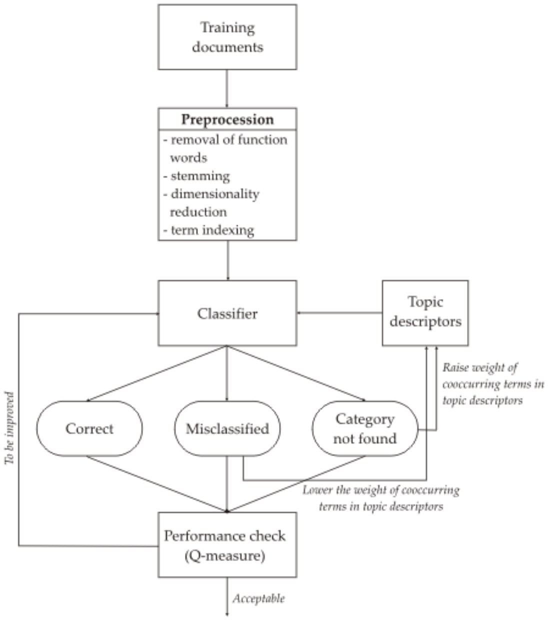
Training documents
Preprocession
- removal of function words
- stemming
- dimensionality reduction
- term indexing
Classifier
Topic descriptors
Raise weight of cooccurring terms in topic descriptors
To be improved
Correct
Misclassified
Category not found
Lower the weight of cooccurring terms in topic descriptors
Performance check (Q-measure)
Acceptable

Figure 1
The flowchart of the training algorithm of UFEX

3.2 Text Categorization Interface of UFEX

HITEC is an implementation of UFEX that is particularly created to handle text documents. Its main goal is to perform text specific processing tasks, that is to transform textual documents to numerical vectors.

The performance of categorization strongly depends on the quality of training data. For efficient training HITEC requires

1 fixed *category system*; during the operational phase the new, “unknown” documents will be classified into that system;

2 and some (min. 5-10) relevant *training documents* for each category of the category system. The necessary number of training documents required for successful training depends on the number of categories and the generality of subject domain. HITEC (and all other supervised learning based categorizers) is unable to learn such categories for which training documents are not provided at all.

The training phase can practically be considered as pre-procession, i.e. it is not a part of the operational phase. It is performed typically once, before the put-on of the classifier. Alternatively, it can be further executed periodically, if more training documents are available (e.g. documents classified by HITEC and checked by an expert) in order to enhance the accuracy of the classifier.

During the operation, HITEC returns the list of most relevant categories for unknown documents. The relevance is expressed by confidence values. The greater is this value HITEC deems the more relevant the corresponding category to the document.

4 Expreimental Results

In this section we present our results achieved with HITEC on RCV1.

4.1 Train/test Split

In our first test we investigated the effect of different train/test splits. The collection was divided into two parts: the first part of the documents was exclusively used to train the neural network of HITEC, while the second part was only used to check to effectiveness of the training on the remaining documentes. We fixed a feature selection setting and only varied the split parameter.

The split was created by determining the ratio (percentage) of the test documentes in the entire collection. That is 90 refers to settings of 10/90 train/test split (see also Figure 2). The training sets were formed on the basis of position of document list. In case of settings 90, every tenth document was taken out for train. With each split we performed 10 cross-validation runs to balance the possible biased selection of training documents. In case of settings 90, it resulted that each document was used at training once, and 9 times at test.

We can observe on Figure 2 that the performance does not raise significantly with the increase of the number train documents. This is due to the fact that only 103 documents are classified only to 103 categories, i.e. even at the 10/90 train/test split, there is about 780 train documents to each category in average. Therefore we can conclude that a small portion of the entire collection is enough for a sufficiently good performance. In other way there is no need to overdose the training documents, and because the time requirement of training heavily depends on the number of training documents, thus we can save considerable time on validation and further the setting of further parameters.

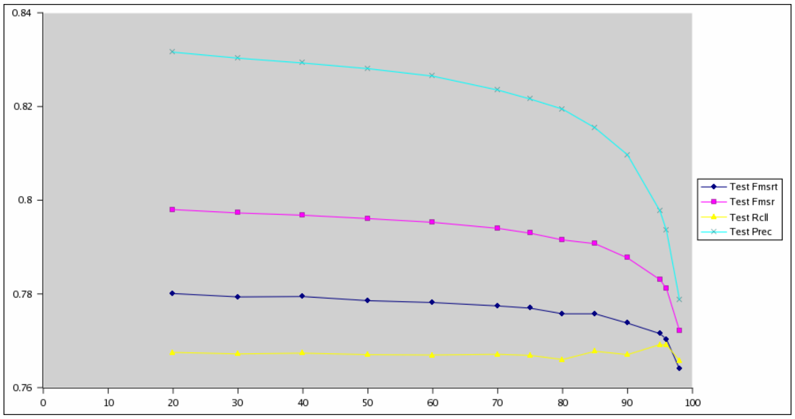

Figure 2

The performance w.r.t to different train/test split. The *x* axis indicates the percentage of test documents, and *y* the value os performance measures.

Instead of the ratio based training set creation, one can choose a different justification of selection by exploiting that RCV1 documents are ordered by publishing time. One can select then the news of the first few days as training set and the remaining days as test set, as done by Lewis at al in [11]. This selection imitates the training and test phase of a corporate document archiving system. This solution has the drawback, however, that news about occasional (Olympics, elections) or unforseen events (9.11 terrorist attacks, threat of bird flu pandemic) may be under or even not represented in the training set.

4.2 Feature Selection

In this test we varied one of the basic thresholding parameters at feature set selection, the required minimal occurrence of a term in the corpus *(d)*. In our test we experienced with three different *d* values: 15, 30, 50. The selection and consequently the size of the feature set has important impact on the running time of the document processing and training. The smaller the size of the feature set, the less calculation is perfomed. Therefore larger *d* values at similar performance are more favorable. Figure 3 shows that in our case the increase of value *d*, that is a smaller dictionary result in similar, or in the case of test setting 95 and 98, somewhat better performance. These results justify the drastic cut at the size of the dictionary.

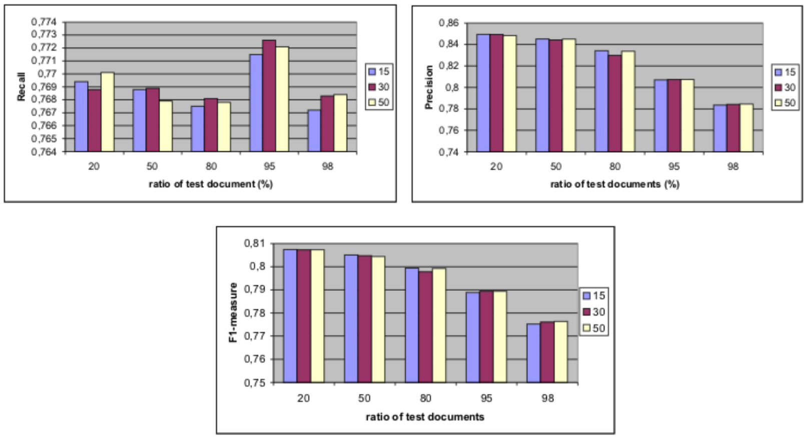

Figure 3

The graphs show the impact of discarding rare terms on the performance measures. The bars indicates when only terms occurring more than 15, 30 and 50 times are kept in the dictionary.

It is interesting to see (Table 1) that though the size of the dictionary decreases significantly by the increase of *d*, but the average frequency size of a document hardly changes. This explains the results achieved concerning the independency of *d* values and performance measures.

Table 1

The impact of different *d* values on the size of dictionary and average frequency size

d	dictionary size	average freq. size
15	66145	69.3
30	47702	68.8
50	37102	68.3

4.3 Parameters of Categorization

The inference algorithm and the balance between evaluation measures precision and recall can be controlled via two main parameters: max-variance (ϑ) and min-threshold (θ). Maximum variance defines the ratio how a node can be differ from the best one to be still selected, while the threshold sets the minimal value of a node to be selected. Obviously, the smaller the value of ϑ, the more nodes are selected and consequently the recall increases and precision decreases. Analogously, the smaller the value of θ, the more nodes are selected at the given level of the taxonomy. Figures 4 and 5 depict the values of evaluation measures in terms of the maximum variance and the minimal threshold, respectively.

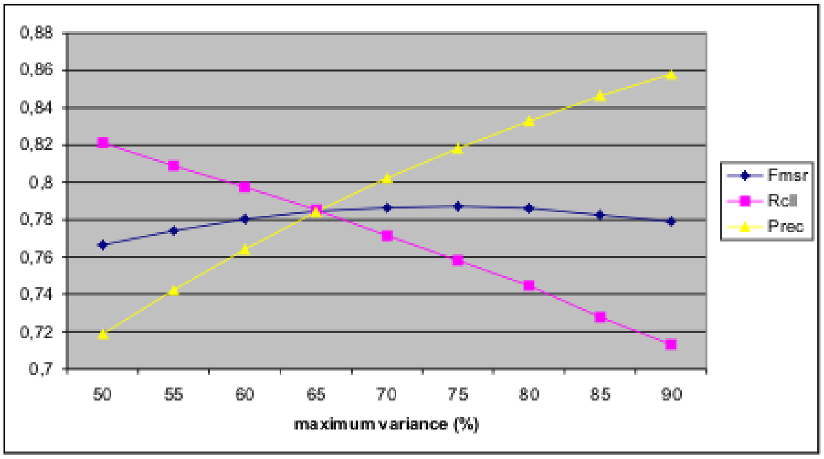

Figure 4

The graphs show the evaluation measures as the function of maximum variance parameter (ϑ)

Based on the Figures one can easily set the break-even point of precision and recall. It is at $\theta = 0.04$ and $\vartheta = 65\%$.

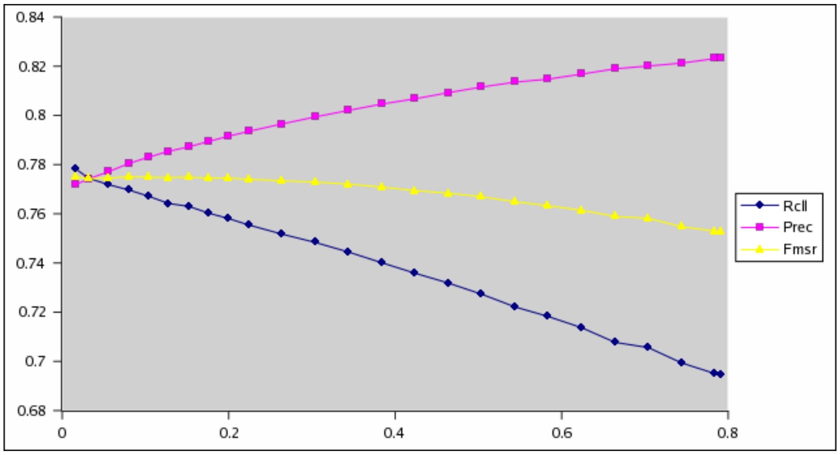

Figure 5

The graphs show the evaluation measures as the function of minimal threshold parameter (θ)

Acknowledgement

Domonkos Tikk was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Science.

Conclusion

In this paper we investigated the performance of HITEC on the new benchmark data collection, the Reuters Corpus Volume 1. We found that our method can achieve comparable to the best known results in the literature, but with a somewhat different setting. Our future work is to run the above tests on completely identical settings.

References

[1] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan: Scalable Feature Selection, Classification and Signature Generation for Organizing Large Text Databases into Hierarchical Topic Taxonomies, in The VLDB Journal, Vol. 7, No. 3, 1998, pp. 163-178

[2] W. Chuang, A. Tiyyagura, J. Yang, and G. Giuffrida: A Fast Algorithm for Hierarchical Text Classification, in Proc. of the 2nd Int. Conf. on Data Warehousing and Knowledge Discovery (DaWaK'00), London-Greenwich, UK, 2000, pp. 409-418, http://www.cs.iastate.edu/~yang/Papers/dawak00.ps

[3] S. D'Alessio, K. Murray, R. Schiaffino, and A. Kershenbaum: The Effect of Using Hierarchical Classifiers in Text Categorazation, in Proc. of 6th International Conference Recherche d'Information Assistee par Ordinateur (RIAO-00), Paris, France, 2000, pp. 302-313, http://133.23.229.11/~ysuzuki/Proceedingsall/RIAO2000/Wednesday/26BO2.ps

[4] D. Koller and M. Sahami: Hierarchically Classifying Documents Using a Very Few Words, in International Conference on Machine Learning, Vol. 14, San Mateo, CA: Morgan-Kaufmann, 1997

[5] A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng: Improving Text Classification by Shrinkage in a Hierarchy of Classes, in Proc. of ICML-98, 1998, http://www-2.cs.cmu.edu/~mccallum/papers/hier-icml98.ps.gz

[6] D. Tikk, G. Biró, and J. D. Yang: Experiments with a Hierarchical Text Categorization Method on WIPO Patent Collections, in Applied Research in Uncertainty Modelling and Analysis, ser. International Series in Intelligent Technologies, N. O. Attok-Okine and B. M. Ayyub, Eds. Springer, 2005, No. 20, pp. 283-302

[7] D. Tikk and G. Biró: Experiments with Multilabel Text Classifier on the Reuters Collection, in Proceedings of International Conference on Computational Cybernetics (ICCC 2003), Siófok, Hungary, August 29-31, 2003, pp. 33-38

[8] W. Wibovo and H. E. Williams: Simple and Accurate Feature Selection for Hierarchical Categorisation, in Proc. of the 2002 ACM Symposium on Document Engineering, McLean, Virginia, USA, 2002, pp. 111-118

[9] F. Sebastiani: Machine Learning in Automated Text Categorization, ACM Computing Surveys, Vol. 34, No. 1, March 2002, pp. 1-47

[10] T. G. Rose, M. Stevenson, and M. Whitehead: The Reuters Corpus Volume 1 – from Yesterday’s News to Tomorrow’s Language Resources, in Proc. of the 3rd Int. Conf. on Language Resources and Evaluation, Las Palmas de Gran Canaria, Spain, 2002, pp. 29-31

[11] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li: RCV1: a New Benchmark Collection for Text Categorization Research, Journal of Machine Learning Research, Vol. 5, 2004, pp. 361-397, http://www.jmlr.org/papers/volume5/lewis04a/lewis04a.pdf

[12] HITEC categorizer online, http://categorizer.tmit.bme.hu

[13] D. Tikk, J. D. Yang, and S. L. Bang: Hierarchical Text Categorization Using Fuzzy Relational Thesaurus, Kybernetika, Vol. 39, No. 5, 2003, pp. 583-600