Case Report ■

Five-way Smoking Status Classification Using Text Hot-Spot Identification and Error-correcting Output Codes

AARON M. COHEN, MD, MS

Abstract We participated in the i2b2 smoking status classification challenge task. The purpose of this task was to evaluate the ability of systems to automatically identify patient smoking status from discharge summaries. Our submission included several techniques that we compared and studied, including hot-spot identification, zero-vector filtering, inverse class frequency weighting, error-correcting output codes, and post-processing rules. We evaluated our approaches using the same methods as the i2b2 task organizers, using micro- and macro-averaged F1 as the primary performance metric. Our best performing system achieved a micro-F1 of 0.9000 on the test collection, equivalent to the best performing system submitted to the i2b2 challenge. Hot-spot identification, zero-vector filtering, classifier weighting, and error correcting output coding contributed additively to increased performance, with hot-spot identification having by far the largest positive effect. High performance on automatic identification of patient smoking status from discharge summaries is achievable with the efficient and straightforward machine learning techniques studied here.

■ **J Am Med Inform Assoc.** 2008;15:32–35. DOI 10.1197/jamia.M2434.

Introduction

Automated document classification can be a powerful technique to aid biomedical researchers by reducing the human effort needed to make repeated decisions categorizing samples of text. It is useful when each of the samples from a given source needs to be categorized into one or more of several pre-defined categories (*labels* or *classes*), and there already exists (or can easily be created) a set of already categorized documents, known as *training data*, usually created by human experts. In this situation, machine learning algorithms can be used to extract a set of rules or procedures from the training data, and apply these rules to new, previously unseen documents, accurately predicting the labels that should be assigned to these documents.

Case Description

The i2b2 challenge modeled the task of identifying patient records of interest in terms of smoking status. The challenge was organized to evaluate automated systems identifying the smoking status of a patient from a hospital discharge summary. Smoking status was defined as one of five mutually exclusive categories: UNKNOWN, NON-SMOKER, SMOKER (current status unknown), PAST SMOKER, and CURRENT SMOKER.

Human experts created training and test sets by manually assigning labels. The task organizers studied the performance of the human annotators during the creation of the test collections. Two human annotators agreed about 81.5% of the time on which one of the five labels to assign to each summary. If automated systems can perform at this level such systems could be useful in a clinical research setting.

Affiliation of the author: Department of Medical Informatics and Clinical Epidemiology, School of Medicine, Oregon Health & Science University, Portland, OR.

Correspondence: Aaron M. Cohen, MD, MS, Department of Medical Informatics and Clinical Epidemiology, School of Medicine, Oregon Health & Science University, 3181 S.W. Sam Jackson Park Road, Mail Code: BICC, Portland, OR, 97239-3098; e-mail: <cohenaa@ohsu.edu>.

Received for review: 03/13/07; accepted for publication: 10/03/07.

Methods

Classifier System Approach

Our approach to this multi-classification problem training uses a sequence of five steps.

Hot-spot Passage Isolation

We hypothesized that words would have different effects on the classification depending upon where they occurred in the discharge summary, and that words occurring near text describing the smoking status of the individual would be the most important. Since our basic classifier approach was to treat features extracted from the text without specific position information ("bag of words"),[1] we needed a simple way to incorporate this into the algorithm.

We found that there were a small number of patterns in the training data that indicated that the nearby text pertained to the patient's smoking status. We called this text "hot-spots," from which we could extract the word-based features. Using cross-validation we found good performance by simply taking a window of text up to 100 characters before and after an identified hot-spot. The hot-spot identified passages were then isolated and passed on to the next step in the process. The rest of the discharge summary was ignored. The hot-spot identifying text patterns that we used are shown in Table 1.

Tokenization and Vectorization

Once the hot-spot identified passages were isolated, these were tokenized into individual words and symbols using the *StandardAnalyzer* module available in the Apache Lucene search engine library (available at http://jakarta.apache.org/

Table 1 ■ Hot-spot Identifying Phrases for Smoking Status Within Discharge Summaries

"smok"	"cig"	"tobac"
"packs"	"tob "	"nicotine"

lucene). This tokenizer handles non-space delimiters well. It also implements a small stop list and filters out the stop words "no" and "not," which we thought might be important for determining smoking status. However, cross-validation experiments on the training data found no differences in performance when these negation words were included or excluded.

After tokenization, the word list was converted into a binary vector. Each position in the vector provides a binary value giving the status of a specific token within the text sample. All token based features that were identified by the hot-spot and tokenization process were included in the feature vectors.

Zero Vector Filtering

After passing a discharge summary through the hot-spot and tokenization processes, some documents resulted in a zero feature vector, with no identified tokens for that sample. These samples will not add useful decision making information to the machine learning algorithms, and may bias the algorithms by contributing to class prior-probabilities.

Filtering out the zero vector samples has to be done both during training and classification of unknown samples. Since the classification task specifically includes an UNKNOWN category, it makes the most sense to classify these samples into this class without using the machine learning model. In effect the system uses the lack of non-zero classification features as the rule for classifying a sample as UNKNOWN.

Multiple Classification Using Error Correcting Output Codes and Support Vector Machines

Error correcting output codes: Error-correcting output codes (ECOC) is a technique used in communications based on information theory that adds redundancy to an encoded bit stream in order to allow detection and correction of incorrectly received bits. The application of this technique to multi-classification problems was first described by Dietterich and Bakiri,[2] and is a powerful way of handling mutually exclusive text multi-classification problems.[3] While it is well known in the machine learning community, it has not been frequently applied in the biomedical informatics domain.

In the ECOC technique a multiple classification problem is reduced to solving multiple binary classification problems by partitioning the original classes into several positive and negative sets. An *ensemble*[4] of classifiers is then trained on the training data, one per partition. When an unknown sample is to be classified, each classifier in the ensemble makes a binary prediction, and then the set of binary predictions is compared to the original partitioning in order to find the original class with the closest binary partitioning. The idea here is to create an error-correcting set of partitions where making a few binary classification mistakes does not prevent the classification ensemble from correctly categorizing a sample into the correct one of several mutually exclusive classes.

The main advantage of the ECOC technique is that it can recover from errors in the individual binary classifiers more gracefully than other popular multi-classification methods such as one-against-all-others. The one-against-all-others method creates a single classifier for each of k classes and uses all samples from the other classes as negatives. Therefore a single classifier error in the classifier of the sample's true class will result in an incorrectly classified sample. For example, assuming that the average probability of error for any single binary classifier in the 5-way problem is equal to p. The 5-way problem uses 15 partitions, each with a mutual Hamming distance of eight, meaning that four binary errors must be made before the sample is incorrectly classified. The probability of an error using the one-against-all-others method on a sample of a given class is then proportional to p, while the probability of an error using ECOC is proportional to p^4 since four errors have to be made before the ECOC classifier ensemble makes an error. Even if the average probability of one of the ECOC ensemble classifiers is larger than p, the ECOC method is likely to show improvement until the average classifier error rate approaches the fourth root of p.

Weighted support vector machines: In practice, any binary classification learning algorithm could be used to implement the individual ECOC binary classifiers. However, we choose to use the libSVM implementation of the linear support vector machine (SVM) technique first proposed by Vapnik.[5,6] SVM performs well on sparse classification problems,[7] and we have had good results with it in the past.[8,9]

One issue with SVM-based classifiers that has sometimes been troublesome is that if one class is more common than another, then SVM will favor the more frequent class. To address this we used the weight parameter available in libSVM. The unweighted approach just has all the weights set to 1.0, for all of the binary classifiers. In the weighted approach, the positive and negative classes for each binary classifier are weighted by their relative rarity using the following formula:

$$w_{class} = (N - N_{class})/N \quad (1)$$

Where N is the total number of samples and N_{class} is the number of samples of the given class. When using the ECOC method, N_{class} is either N_+, the number of samples in the positive partition for a given binary classifier, of N_-, the number of samples in the negative partition.

Post-processing Rules

We hypothesized that there might not be enough data for the algorithm to identify all important combinations of features. Using cross-validation on the training set, we examined the errors that our system made and created post-processing transformation rules based on the predicted class of the ECOC ensemble and text patterns that we found during the error analysis. If the sample had a prediction that matched the text pattern assigned to the rule then the predicted class was modified according to the rule. For example, one of our rules was if SMOKER was the predicted class, and the text "no history of smoking" was found, then change prediction to NON-SMOKER. Since the rules were derived by manual inspection of the training data we could

Table 2 ■ Micro- and Macro-averaged F1 Results for 5-way Classification by Tested Systems on Test Collection

System	Hot-spot	Tokenization	Zero Vector Filtering	SVM Weighting	ECOC	Post-processing Rules	Micro-F1	Macro-F1
Run1	Yes	Lucene	Yes	Yes	Yes	Yes	0.8804	0.6430
Run2	Yes	Lucene	Yes	Yes	Yes	No	0.8860	0.6504
Run3	Yes	Lucene	Yes	No	Yes	Yes	0.8713	0.6190
A (not submitted)	Yes	Simple	Yes	Yes	Yes	No	0.9000	0.6780
B (not submitted)	Yes	Lucene	Yes	No	Yes	No	0.8720	0.6220
C (not submitted)	Yes	Lucene	No	No	Yes	No	0.8470	0.5960
D (not submitted)	Yes	Lucene	No	Yes	Yes	No	0.8340	0.5780
E (not submitted)	Yes	Simple	No	Yes	Yes	No	0.8660	0.6250
F (not submitted)	Yes	Lucene	Yes	No	No	No	0.8380	0.5640
G (not submitted)	Yes	Lucene	Yes	Yes	No	No	0.8660	0.6090
H (not submitted)	Yes	Lucene	No	No	No	No	0.8360	0.5670
I (not submitted)	No	Simple	Yes	Yes	Yes	No	0.6040	0.3050
i2b2 Best							0.8955	0.7568

not accurately estimate their effect on performance before using them on the test data.

Evaluation

The i2b2 challenge task was evaluated using the standard classification measures of precision, recall, and F1-measure, extended for multiple classification problems.[1] The primary measure of comparison was the F1-measure, with precision and recall weighted equally (beta = 1.0). Micro- and macro-averaged F1 values were computed.

For the i2b2 smoking status classification task, the organizers applied micro- and macro- averaged F1 in two ways. Submissions were scored as 5-way tasks as given in the training data, with each sample being classified into one of the five smoking status categories. They also created a 3-way task post-hoc, grouping SMOKER, PAST SMOKER, and CURRENT SMOKER into a single one SMOKER class.

Results

In addition to the officially submitted and scored runs, we conducted a series of experiments to make it possible to determine which features of our system lead to improved performance. Table 2 shows the results of these experiments for the 5-way classification problem, and Table 3 shows the performance of our submissions on the 3-way task.

The systems labeled Run1-3 were our officially submitted runs, the systems labeled A-I are additional system configurations scored using methods equivalent to those used by the task organizers. The system labeled "i2b2 Best" was the best performing system run submitted to the task organizers. Configurations A-I systematically vary properties of our classification approach making it possible to determine the contribution to performance for each property.

As seen in Table 2, Run2 was our best submitted run, using Lucene tokenization, zero-vector filtering, SVM weighting, and ECOC, but not the post processing rules. This run scored second out of all runs submitted. Though we did not see this effect on our cross-validation experiments before submitting our official runs, on the test data there is degradation in performance when using stop word filtering. Comparing system A to Run2, the only difference is the tokenization and lack of stop word filtering in system A. System A has a micro-F1 of 0.9000, which is 0.0140 better than Run1, actually higher than the best performing system submitted to the i2b2 challenge task.

By far the largest effect was due to the hot-spotting technique. This can be seen by comparing the results of System A with System I, hot-spotting being the one difference between these approaches. The micro-F1 difference is huge, 0.2960, with the hot-spotting system achieving a micro-F1 of 0.9000 and the non-hot-spotting system only 0.6040. Simply removing hot-spotting transforms the best performing system presented here into the worst.

Table 2 includes several other informative comparisons. The difference between Run2 and E is the use of zero-vector filtering. Run2 outperforms E by 0.02. SVM weighting is the difference between Run2 and B, and Run 2 outperforms B by 0.014. The ECOC technique is compared to the one-against-all-others in Run1 versus system G. This time Run1 outperforms G by 0.020. For the 5-way evaluations, the use of the post-processing rules was counterproductive. The use of this technique is the only difference between Run1 and Run2, and Run2 does a little better without it.

Finally, System H uses the inferior choice for each of the three alternative techniques (zero-vector filtering, weighting, ECOC), using hot-spotting but no post-processing rules (as with our best system), and achieves a micro-F1 of 0.8360, which is 0.0640 less than the best performing system. Interestingly this is very close to the sum of the individual contributions of each of the left-out features: 0.0140 + 0.20 + 0.0140 + 0.20 = 0.0680.

Table 3 ■ Micro- and Macro-averaged F1 Results for 3-way Classification by Tested Systems on Test Collection

System	Micro-F1	Macro-F1
Run1	0.9516	0.9136
Run2	0.9615	0.9317
Run3	0.9713	0.9493
i2b2 Best	0.9713	0.9518

Discussion

By far, the biggest difference in performance was due to hot-spot filtering of the discharge summary text. The hot-spotting technique was a simple way to determine what areas of text to focus on, and to filter out a large amount of noise. Furthermore, the hot-spotting technique was completely reliable for this task using only the six identifying

phrases shown in Table 1. For systems implementing hot-spotting and zero-vector filtering, in our experiments with the test collection, precision and recall of the UNKNOWN class were both 100%.

Besides hot-spotting, the other techniques proposed and used here, including zero vector filtering, SVM inverse proportional weighting, and the use of ECOC for multi-classification problems all add a noticeable improvement to the system performance. While it was unexpected that these techniques would additively combine to produce an overall linear increase in performance, this does appear to be approximately the case.

The post-processing rules did not help performance. Nevertheless, one of our systems that included these rules, Run3, was the top performing submitted system when evaluated on the Micro-F1 on the 3-way task. Since systems were not optimized for the 3-way problem, it is not clear that these results are meaningful for comparison purposes across submissions. However, the results are meaningful for understanding how well systems can do when training on 5-way data for tasks where the 3-way labeling is important. At micro-F1 measures of over 0.97, it is clear that the top performing submissions can assign 3-way smoking status with performance adequate for many practical purposes.

We have shown the effectiveness of several general techniques that are particularly well suited to the 5- and 3-way smoking status multi-classification task. The techniques are of modest implementation complexity, highly runtime efficient, and based on solid theory and straightforward assumptions. Evaluation on independent test data shows performance competitive with the best available techniques, including more complex NLP-based methods and additional customized training and lexicon data. Further work will study the level of performance required for specific applications, the effectiveness of these systems to support specific areas of biomedical research. An expanded version of this paper is available as a JAMIA on-line data supplement at www.jamia.org.

References ■

1. Sebastiani F. Machine learning in automated text categorization. ACM Computing Surveys (CSUR). 2002;34:1–47.
2. Dietterich TG, Bakiri G. Solving multiclass learning problems via error-correcting output codes. J Artif Intell Res. 1995:263–86.
3. Ghani R. Using error-correcting codes for text classification. In: Langley P, editor. Proceedings of {ICML}-00, 17th International Conference on Machine Learning; 2000: Morgan Kaufmann Publishers, San Francisco, US; 2000. pp 303–10.
4. Dietterich TG. Ensemble methods in machine learning. Lecture Notes in Computer Science. 2000;1857:1–15.
5. Chang C-C, Lin C-J. LIBSVM: a library for support vector machines, 2001; Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. Accessed March 20, 2006.
6. Vapnik VN. The nature of statistical learning theory. 2nd ed. New York: Springer; 2000.
7. Joachims T. Text categorization with support vector machines: learning with many relevant features. Proc 10th Eur Conf Mach Learn. 1998;137–42.
8. Cohen AM. An effective general purpose approach for automated biomedical document classification. AMIA Annu Symp Proc. 2006:161–5.
9. Cohen AM, Yang J, Hersh WR. A Comparison of Techniques for Classification and Ad Hoc Retrieval of Biomedical Documents. Proc Fourteenth Annu Text REtrieval Conf - TREC 2005; Gaithersburg, MD; 2005.